Understanding Role of Provenance in Bioinformatics Workflows and Enabling Interoperable Computational Analysis Sharing

Farah Zaib Khan

Submitted in total fulfillment of the requirements of the degree of

Doctor of Philosophy

School of Computing and Information Systems

The University of Melbourne

December 2018

Abstract

The automation of computational analyses in data-intensive domains such as genomics through scientific workflows is a widely adopted practice in many fields of research nowadays. Computationally driven data-intensive experiments using workflows enable Automation, Scaling, Adaption and Provenance support (ASAP). Provenance data collection is an essential factor for any computational workflow-centric research to achieve reproducibility, transparency and support trust in the published results. At present capture of provenance information across the plethora of workflow management systems and custom software platforms in the bioinformatics domain is not well supported and as such, there exist numerous challenges associated with the effective sharing, publication, understandability, reproducibility and repeatability of scientific workflows.

This thesis focuses on providing a unified, interoperable and systematised view of provenance with specific focus on workflow environments in the bioinformatics domain. We identify and overcome the current disconnect between various workflows systems and their existing provenance representations. Through empirical analysis of complex genomic data analysis workflows using three exemplar workflow systems, we identify implicit assumptions that arise. These assumptions produce an incomplete view of provenance resulting in insufficient details that impact on workflow enactment requirements and ultimately on the

reproducibility of the given analysis. We propose a set of recommendations to mitigate against such assumptions and enable workflow systems to document and capture complete provenance information that can subsequently be used for re-enacting workflows in other contexts and potentially using other workflow platforms.

Based on this empirical case study and pragmatic analysis of related literature, we define a hierarchical provenance framework offering *"Levels of Provenance and Resource Sharing"*. Each level of this framework addresses specific provenance recommendations and supports the capture of rich provenance information, with the topmost layer enabling the sharing of comprehensive and executable workflows utilising retrospective provenance. To realise this framework, we leverage community-driven, domain-neutral, platform-independent and open-source standards to implement **"CWLProv"** - a format for the methodical representation of provenance supporting workflow enactment aggregating resources specific to the given enactment and associated workflow configuration settings. We realise CWLProv through the Common Workflow Language (CWL) for workflow definition and utilise Research Objects (ROs) for resource aggregation and PROV-Data Model (PROV-DM) to support the capture of retrospective provenance information as required for subsequent workflow enactments.

To demonstrate the applicability of *CWLProv*, we extend an existing workflow executor (cwltool) to provide a reference implementation that generates metadata and provenance-rich interoperable workflow-centric ROs. This approach aggregates and preserves data and methods needed to support the coherent sharing of computational analyses and experiments. Evaluation of CWLProv using real-life bioinformatics pipelines is demonstrated to highlight the utility of the approach demonstrating the interoperability of workflow analyses and the benefits to research reproducibility more generally.

DECLARATION

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Farah Zaib Khan, December 2018

Acknowledgements

The motivation to begin my PhD journey, the morale to keep going and the encouragement to complete it –all were borrowed from or gifted by the unbelievably supportive individuals in my life for whom I owe a debt of gratitude

I am grateful to ALLAH Almighty for blessing me with more than I deserve and for presenting me with more opportunities that one could ever dream of.

My deepest gratitude goes to my supervisors A.Prof Andrew Lonie and Prof. Richard Sinnott for accepting me as their student to undertake this PhD and for the guidance, support, advice and opportunities they provided throughout my candidature. These few words can never do justice to the continuous mentoring, timely feedback and constant encouragement they provided, without which it would have been impossible to develop the research skills required to complete this journey.

I would also like to thank my collaborators Stian Soiland-Reyes and Michael R. Crusoe for their valuable feedback, insightful comments on my research, stimulating discussions and above all for being there whenever I needed help. I wish to thank The University of Melbourne for the scholarship awarded to me to carry out this research. Thanks to the staff at UniMelb especially Rhonda Smithies & Dr. Antonette Mendoza for always listening to my concerns and offering help.

I would like to extend my sincere gratitude to my mentor Prof. Raheel Qamar for always believing in me and encouraging me to apply for the higher studies in the first place. I owe him much more than I can express in these few words. I wish to thank the friends in Melbourne who were my family away from home throughout my PhD journey. Neelofar, Sehrish & Shafqat! You kept my morale high in my time of need and shared laughter in times of happiness. Thank you for being true and honest friends.

My uttermost gratitude to my loving family, without whom I would not be the person I am today. Abbu for making me strong and independent, for trusting me with everything, for showing confidence in me and always encouraging me to broaden my horizons. Thank you Amma for teaching me selflessness and the devotion that one should have to their cause; for dedicating your life to your kids and for the continuous love, respect & appreciation you have showered us with. Special thanks to my sisters Neelam, Anam & Summaiya and my little brothers Osama & Hamza for being understanding and encouraging me in all my pursuits.

Finally, my dearest husband Ahsen! Thank you for giving me a reason to look forward to every December and the motivation to work for the rest of the year; for giving me strength every day even though we are 6 hours & 9,230 km apart; for signing up for this journey with me; for telling that you are proud of me and waiting for me patiently while I chase my dreams. Thank you for your faithful companionship and for supporting me through thick and thin for the past seven years. Because of your efforts and patience we are together. Thank you for everything!

PREFACE

This thesis research has been carried out at School of Computing & Information Systems, The University of Melbourne. The contributions of this thesis are discussed in Chapter 2 - 6 which are based on the following publications.

Publications from the PhD Research

- Published: Kanwal, Sehrish and **Khan, Farah Zaib** and Lonie, Andrew and Sinnott, Richard O. **Investigating reproducibility and tracking provenance A genomic workflow case study.** In: BMC bioinformatics 18.1 (2017), p. 337 (doi: 10.1186/s12859-017-1747-0)
  First and Second author contributed equally.

- Submitted: **Khan, Farah Zaib** and Soiland-Reyes, Stian and Sinnott, Richard O. and Lonie, Andrew and Goble, Carole and Crusoe, Michael R. **"Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv".** In: GigaScience (doi: 10.5281/zenodo.1966881)

Poster, Abstract Submissions and Presentations

- Abstract & Long Talk: Kanwal, Sehrish and **Khan, Farah Zaib** and Lonie, Andrew and Sinnott, Richard O. **"Reproducibility and Provenance Track-**

ing of Genomic Workflows" In eResearch Australasia Conference, 2016, Melbourne, Australia

- Poster: Kanwal, Sehrish and **Khan, Farah Zaib** and Lonie, Andrew and Sinnott, Richard O. **"Reproducibility of Computational Genomic Workflows"** In: Galaxy Australasia Meeting, 2017, Melbourne, Australia

- Abstract & Poster: **Khan, Farah Zaib** and Soiland-Reyes, Stian and Lonie, Andrew and Sinnott, Richard O.
  CWL+Research Object == Complete Provenance In: Bioinformatics Open Science Conference (BOSC), 2017, Prague, Czech Republic
  (doi: 10.7490/f1000research.1114781.1)

- Abstract & Long Talk: **Khan, Farah Zaib** and Soiland-Reyes, Stian and Sinnott, Richard O. and Lonie, Andrew and Crusoe, Michael R. **CWLProv –Interoperable retrospective provenance capture and its challenges** In: Galaxy Community Conference & Bioinformatics Open Science Conference (GCC-BOSC), 2018, Portland Oregon, USA
  (doi: 10.7490/f1000research.1115721.1)

- Abstract & Long Talk: Soiland-Reyes, Stian and **Khan, Farah Zaib** and Sinnott, Richard O. and Lonie, Andrew and Crusoe, Michael R. and Goble, Carole. **Capturing interoperable reproducible workflows with Common Workflow Language** In Workshop on Research Objects (RO 2018), Amsterdam, Netherlands.
  (doi: 10.5281/zenodo.1312623)

- Lightning talk: Khan, Farah Zaib et al. **CWLProv –Towards Reproducible & Interoperable Scientific Workflows** In: From replication crisis to credibility revolution, 2018, Melbourne, Australia

Open Source Project Contribution

Active member of the community-driven project “Common Workflow Language (CWL)” (https://www.commonwl.org/) and devised standardisation for the CWL workflow enactment & its provenance as part of this thesis research.

*To **my father***

The strength of my life who always believed in me & encouraged me to be the best version of myself

*To **my mother***

The hope of my life who devoted her life to me & was selflessly there whenever I needed her

*To **my husband***

The love of my life who made me believe in true love & stood by my side through the toughest times

ABBREVIATIONS

Common Workflow Language	CWL
Directed Acyclic Graph	DAG
Research Object	RO
Genome Analysis Toolkit	GATK
Genomics Virtual Lab	GVL
Next Generation Sequencing	NGS
High Performance Computing	HPC
Operating System	OS
Workflow Management System	WMS
National eResearch Collaboration Tools and Resources	NeCTAR
PROVenance Data Model	PROV-DM

CONTENTS

List of Figures

List of Tables

Chapter 1

Introduction

In describing the goals of the scholarly communication of scientific research via publications, Mesirov [1] said:

"Scientific publications have at least two goals: (i) to announce a result and (ii) to convince readers that the result is correct."

Convincing readers to accept results as presented, without providing the means to validate the experiment or verify the results, is at best an inequitable expectation. Ideally, given a scientific publication, a reader should be able to verify the presented claims seamlessly using the information provided by the authors. However, establishing trust of experiments and their outcomes is subject to levels of transparency maintained in the communication of the methodology employed in the reported experiments. Transparent and comprehensive communication of the research process followed in any scientific publication requires documenting information that can answer questions such as "what was used to generate this output?", "where did this data come from?", "why was a particular method preferred over others?" and "How was the output generated?". Information providing answers to such fundamental queries is known as *provenance* (see the more formal definition in *Chapter 2*).

Dataflow-oriented scientific workflows are proposed [2] and have been actively used for data-intensive *in silico* experiments making them an integral part of the scholarly knowledge-cycle [1]. Such workflows enable **A**utomation, **S**caling, **A**daption and **P**rovenance support (**ASAP**) [3] and ideally be *"key enablers for reproducibility of experiments involving large-scope computations"* [4]. With the extensive use of workflows as computational methods, many Workflow Management Systems (WMSs) have been designed for systematising the representation and management of complex computational experiments comprised of inter dependent data analysis tasks. Along with various advantages such as efficient task scheduling, data management, modular methods and improved debugging, such systems are expected to enable the automated capture of provenance information (data) to document data dependencies and the derivation process. There exist a wide range of domain-agnostic WMS [5, 6, 7] that address the needs of different scientific communities and domain-specific systems such as Galaxy [8] and GenePattern [9] for *-omics* data analysis.

The recent and rapid evolution in the field of genomics, driven by advances in massively parallel DNA sequencing technologies combined with the uptake of genomics as a mechanism for clinical genetic testing, have resulted in high expectations from clinicians and the biomedical community at large regarding the reliable, reproducible, effective and timely use of genomic data to realise the vision of personalised medicine and for improved understanding of various diseases. There has been a contemporaneous upsurge in the number of techniques and platforms developed to support genomic data analysis [10] ,and computational bioinformatics workflows consisting of community generated tools [11] and libraries [12, 13] are often deployed to deal with the data processing requirements. It has therefore become crucial to evolve and optimise Next Generation Sequencing (NGS) data processing and analysis pipelines (workflows) to keep pace with exponentially increasing amounts of genomics data being produced. The ability

to produce digital forms of DNA sequences and other *-omics* data, has outrun the ability to store, transmit and interpret such data. Hence, the major bottleneck in complex experiments involving NGS data is the data processing and associated knowledge generation.

Although there is a large amount of published literature on the use and importance of *-omics* data in research, far less exists on translation into clinical settings [14]. The committee on the review of *-omics*-based tests for predicting patient outcomes in clinical trials [15] attributed this discrepancy to two primary causes: inadequate design of pre-clinical studies and weak bioinformatics rigour. The scientific community has paid special attention to benchmarking *-omics* analysis for quality assurance, analytical validity and accuracy of bioinformatics pipelines [16]. Efforts to define quality standards, good practices, well-articulated checklists and reference materials (e.g. reference data, methods and standards) [15, 17, 18] have been proposed to improve the quality of bioinformatics experiments. However, incomplete provenance documentation of the computational analyses processing *-omics* data, e.g. lacking the information about software names, versions and associated parameter settings adversely impact the interpretation, transparency and reproducibility of these experiments [19, 20].

Despite wide recognition of the need for sharing comprehensive provenance information on experiments, scientific workflows often suffer from what is commonly known as *workflow decay* [21], referring to "broken" workflows, i.e. they are either not comprehensible or reproducible. Four important aspects are considered responsible for workflow decay: volatile third party resources; missing example data; missing execution environment information; and a lack of description of the workflows themselves [22]. With the exception of volatile third party resources, comprehensive provenance information can, at least in principle, be used to tackle the other identified workflow decay issues. Thus capturing com-

plete provenance documentation including the availability of resources initially used to generate a set of results/claims is of paramount importance to scientific rigour.

Other challenges that often occur during the validation/verification of scientific claims lie in restricted or no access to the primary data [23], underlying software and associated computational artefacts including the workflow specifications; limited means to reproduce the workflow platform/environment; reliance on proprietary software; and unstructured organisation of associated artefacts. If/when shared these often have little or no annotations and/or descriptions to help guide future users. Therefore, it is imperative to devise formal strategies for the effective communication of computational experiments.

1.1 Problem Statement

Given a published research paper based on a scientific workflow, the expectation is that end-users should be able to build new research by exploiting sufficient provenance information including systematic methods and associated data documented in that paper. This should allow end-users to fully understand the methods/data-derivation, verify results if need be or re-use the resources for related research objectives. These expectations naturally extend to assuming that end-users can effectively make use of workflow provenance to address queries about the whole process that transformed a particular input dataset into one or many outputs reported in the paper. Often lack of understanding and documentation of crucial information results in coarse-grained and incomplete provenance capture, rendering it insufficient to convey the entire research process.

Another layer of complexity is due to the proliferation of many WMSs de-

signed to address specific compute and data requirements of diverse research communities. A plethora of approaches have been taken to design and implement computational analyses and sharing mechanisms resulting in diverse and heterogeneous platforms, hence it can be a challenge to immediately comprehend published workflow related artefacts. The situation has been exacerbated due to lack of a unified mechanism to enforce the adoption of existing efforts and standards as well as best practices associated with workflow-centric study designs and their communication. This results in inconsistent granularity and typically, poor quality workflow provenance information. The repercussions of this situation include inadequate understanding of research processes/analysis, resulting in a lack of reproducibility and hence compromising the validity of published scientific claims.

Due to the limited adoption of standardisation practices and the absence of agreement on methods to design or publish new research, the already demanding data and compute-intensive *in silico* workflow-centric bioinformatics research is complicated further. Analysing exponentially growing *-omics* data through workflows requires an in-depth understanding of the impact of particular tools, configuration settings, underpinning infrastructure and associated datasets. The heterogeneity of WMSs, customised logging of provenance information, insufficient information shared about crucial domain-aware decisions and limited access to resources utilised makes the understanding and reproducibility of a given bioinformatics analysis extremely difficult and often impossible. As a result, the ability to make new scientific discoveries through extending existing experiments and their experimental infrastructure is seldom fulfilled [19].

To address WMS heterogeneity issues, there have been several initiatives focused on standardisation of individual components/aspects of workflow analyses focused on promoting consensus and interoperability of workflow-based

analyses. Some notable efforts to standardise provenance information and representation include the Open Provenance Model (OPM) [24] and the World Wide Web Consortium PROV Data Model (W3C PROV-DM) [25]. These efforts however are not adopted by all the WMSs and the vast majority of existing systems generate customised logs associated with their workflow enactment. Similarly, efforts for richly annotated representation and preservation of methods and data including the Open Archives Initiative Object Re-use & Exchange (OAI-ORE) [26], DataOne Packages [27], ReproZip [28] and Research Objects (RO) [29] exist but are not (yet) widely adopted.

An open problem in leveraging existing initiatives to produce research outputs is in aggregating methods, data and detailed provenance of analyses such that any outputs are understood and can be used across different WMS and different computing platforms. Due to the specificity and heterogeneity of workflow descriptions, such standardisation efforts are still largely isolated. At present no inter-connected and consolidated solution for workflow analysis representation and dissemination has been widely endorsed and adopted.

Having consensus on a single WMS, a single standard or an aggregation approach for publishing results is unlikely. There are many reasons for this. Each approach has its own pros and cons which makes it suitable for specific cases. The scientific research communities from different domains have adopted their own solutions and frameworks for workflow management. As a result, many solutions exist addressing different problems individually leading to division of efforts from the scientific community. It is highly desirable to develop an abstract layer on top of the existing solutions that is interoperable enough so that the contents of research represented using this abstraction can be understood and utilised across various WMS on different underpinning platforms.

1.2 Research Objectives

The overall aim of this thesis is to facilitate bioinformatics workflow-centric research understanding, improve reproducibility and enable re-use by analysing existing workflow definition approaches and identifying the fundamental elements of workflow provenance information. Building on this, we aim to devise techniques supporting the complete and transparent communication of research. More specifically the objectives of this research are:

O1: To investigate how various workflow definition approaches are addressing the key aspects of provenance;

O2: To identify implicit and explicit assumptions of workflow definition approaches that lead to incomplete provenance documentation;

O3: To develop an understanding of the essential factors/artefacts/resources that must be captured as part of provenance in light of the identified assumptions in O2;

O4: To extend the understanding of such resources from O3 by revisiting literature dedicated to designing best practise recommendations for workflow-centric research and compiling these recommendations;

O5: To devise a conceptual provenance framework utilising the compilation done in O4 to capture the fundamental artefacts identified in O3 and O4 in a hierarchical fashion;

O6: To demonstrate the working of the framework from O5 by formulating a standardised format for interoperable workflow-centric analysis representation, and

O7: To assess the effectiveness of the solution achieved after meeting with real-world bioinformatics workflows.

These objectives translate into three major lines of work supporting the improved understanding and capture of workflow provenance. Firstly, the identification and understanding of **key artefacts of bioinformatics workflow provenance** that must be captured. This results in detailed (white-box) provenance capture during workflow enactments.

Secondly, a **theoretical/conceptual provenance guide** that can serve as a principled approach to be used by researchers for improving the state of workflow provenance and reproducibility in the design process of studies as well as in the analysis process of published studies.

Thirdly, the **standardisation of research representation and sharing** is closely related to the conceptual provenance guide. A standardised and unified format for workflow-centric analyses representation should result in enhanced interoperability and understandability of workflow studies across various platforms.

These objectives are revisited in the final chapter of the thesis (Chapter 7) to assess the extent that this thesis has met these objectives and hence the contribution made.

1.3 Research Questions & Research Activities

This section presents the main hypothesis that motivates this thesis and drives the subsequent research questions and methods adopted. The overarching hypothesis is:

Declarative and platform-agnostic workflow definitions with high degrees of abstraction that leverage standardised and provenance-aware resource aggregation can help address the reproducibility and interoperability issues present in workflow-centric scientific research.

Subsequent research questions are investigated and used to evaluate this hypothesis, considering the three major lines of work identified in Section 1.2. Limited understanding of the implications of provenance completeness, results in coarse-grained and insufficient provenance knowledge that impacts on the sharing and reproducibility of workflow-centric experiments. It is vital to identify the approaches taken by different WMSs to handle provenance capture and identify the critical factors that need to be documented as part of workflow provenance. This brings us to the two research questions, the first is:

RQ1: *How do existing WMSs handle different aspects of governance?*

To answer this question, we investigated the provenance literature and devised a taxonomy. In addition, we categorised the existing workflow definition approaches generally employed to design bioinformatics workflows. Both classifications help in evaluation of the existing WMSs with respect to their provenance capabilities. As evident from the literature, there is inconsistent granularity in provenance offered by many WMSs, due to limited understanding of essential provenance elements; this motivated us to investigate the second research question:

RQ2: *What are the key artefacts that must be documented for comprehensive provenance capture in bioinformatics workflows?*

To address this question, we first identify the implicit assumptions of different workflow definition approaches that can lead to incomplete provenance doc-

umentation. Through such assumptions, we identify key resources and artefacts of bioinformatics workflows, and propose recommendations to capture fine-grained provenance of bioinformatics workflows. These recommendations are based on an empirical case study, however we identify that there are several related studies focused on addressing the similar problem, i.e. complete communication of scientific analyses. To extend our understanding of the crucial artefacts that must be captured in provenance, we consider existing literature focused on defining best practices and recommendations to improve the design of workflow-centric studies by focusing on various aspects of their provenance capture, accessibility of the utilised resources and standardisation support. Having identified the key artefacts of provenance, we attempt to resolve the third core research question:

RQ3: *How can we devise a hierarchical provenance framework encompassing community experiences that can serve as a guiding principle to determine the state of provenance of a given published analysis designed using a particular workflow definition approach?*

This research question focuses on devising a conceptual incremental provenance framework based on the understanding developed in addressing RQ2. In addressing RQ2, we identify that declarative workflow definition approaches make the least assumptions on their execution environment and provide constructs to facilitate fine-grained provenance capture. The conceptual provenance framework resulting from RQ3 and the findings from RQ2 are the driving factors that shape our work on a standard format for the representation of provenance enabled workflow-centric analysis. This brings us to our fourth and final research question:

RQ4: *How can we leverage existing abstraction and standardisation techniques to realise*

the provenance framework and demonstrate its utility?

We will revisit these research questions in the final chapter of this thesis (*Chapter 7*) when the contributions of this thesis are mapped to the research questions. In terms of the signposting of the thesis itself, we performed the following research activities in this thesis to tackle the above stated research questions:

RA1: Literature Classification - We provide a detailed assessment and review of the current research literature focused on categorising the related concepts of provenance, workflow definitions and best practices to design and share workflow-centric studies. This includes:
RA1(a) - Taxonomy definition covering concepts related to provenance that are re-visited in the thesis several times.
RA1(b) - Classification of workflow definition approaches covering three major approaches due to the diversity and heterogeneity of environments. This classification covers the broad spectrum of existing systems with respect to workflow design and the underlying assumptions of each category.
RA1(c) covers a compilation of best practices carried out by pragmatically analysing existing studies that have the goal of improving workflow-centric study design and dissemination.

RA2: Comparative Literature Survey - This thesis provides a systematic and comparative survey of exemplar workflow definition approaches and WMS with respect to their provenance support including mechanisms in place (if any) for facilitating the sharing of related workflow resources. This survey includes the analysis of the state of the art WMSs extensively used in bioinformatics studies and related efforts that aim to achieve similar objectives as described in Section 1.2.

RA3: Empirical Case Study - We implemented a complex variant calling workflow in systems exemplifying each category of workflow definition approach, to understand the implicit details that are often considered too obvious to be documented yet result in reproducibility challenges. These intricate underlying details are referred to as *assumptions* of each approach in this thesis. They have direct implications on the completeness of workflow provenance.

RA4: Theoretical/Conceptual Modelling - We explore existing literature and best practices and recommendations from the scientific community (RA1(c)) to establish a conceptual provenance framework. The hierarchical provenance framework is established in light of the experiences of existing research dedicated to improving workflow development and computational analysis sharing, and specifically to ensure that it is not tied to one specific execution environment or approach.

RA5: Standardisation - We utilise well-defined and community-driven initiatives to devise a standard format for the representation of workflow analysis. This format inherits key characteristics of the underlying standards such as abstraction, domain-independence, support for rich annotations and standardised descriptions.

RA6: Practical Realisation - We extend an existing workflow executor to practically demonstrate the realisation of the conceptual framework devised in RA4 using the standard format developed in RA5.

RA7: Case-based Evaluation - Finally we use three real-life bioinformatics workflows designed by independent research groups to showcase the syntactic, semantic and pragmatic interoperability of the workflow-centric analyses that is supported.

Table 1.1: Methods used in this thesis to address the Research Questions

Research Activities	**RQ1**	**RQ2**	**RQ3**	**RQ4**
Literature Classification	✓	✓	✓	
Comparative Literature Survey	✓			✓
Empirical Case Study		✓	✓	
Theoretical/Conceptual Modelling			✓	
Standardisation				✓
Practical Realisation				✓
Case-based Evaluation				✓

The mapping of these research activities to the corresponding research questions is summarised in Table 1.1.

1.4 Scope of the Study

Provenance of scientific workflows has been an active topic of discussion and exploration over the past decade with various collective community-driven initiatives focused on understanding the many different aspects of provenance. These efforts explore topics such as provenance capture, persistent storage, analytics including querying/visualisation, standardised representation, and provenance sharing [30]. In addition, many different classes of provenance as described in Section 2.2.2 exist in the case of workflow provenance. In this thesis, we focus on the capture of provenance data related to workflow enactment, which is referred to as "Retrospective Provenance" (formally defined in Section 2.2.2.2). In the later chapters, the thesis also briefly discusses aspects of "Prospective Provenance" especially when identifying key artefacts of workflow provenance in empirical and pragmatic analyses.

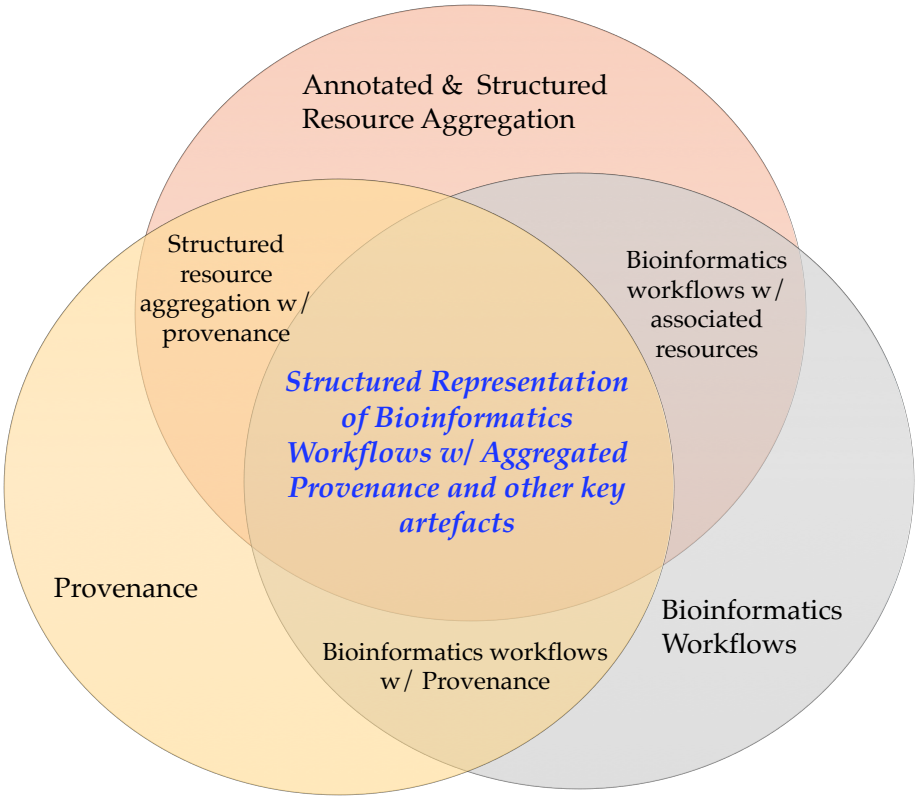

Figure 1.1: Venn Diagram highlighting the area of research with the innermost subset indicating the scope of this thesis

The second important dimension defining the scope of this thesis is the domain of research. This thesis focuses primarily on improving the understanding of provenance with respect to bioinformatics workflows. Various technologies such as the Common Workflow Language (CWL) [31], Research Objects, the Provenance Data Model (PROV-DM) [32] and Provenance Specifications [33] are exploited in devising the solution to bioinformatics workflow provenance capture and representation. For the most part, these are domain-agnostic and can be utilised in any scientific workflow. However, the requirements analysis undertaken in this thesis is based on use of empirical and pragmatic analysis of bioinformatics workflows and bioinformatics research publications only. Figure 1.1 shows the three main focus areas of this thesis discussed in this section.

The third focal point is devising a mechanism to systematically organise and communicate workflows and all related artefacts identified as crucial elements for understanding an analysis, to enable re-using individual components or the whole workflow, and most importantly for artefacts that are essential for reproducing a given analysis. A provenance trace as a document alone is of limited utility with limited or no access to the resources required to enact the workflow. Therefore, the third area of focus is producing structured and annotated aggregations of key resources that allow end users to benefit from published research results based on workflows as computational methods.

The ultimate goal of this thesis is: to improve the provenance documentation of bioinformatics workflows by understanding and characterising key resources; to propose a conceptual provenance framework which helps the research community differentiate between different levels of provenance satisfied by authors when publishing studies, and to devise experimental demonstrations applying the provenance framework using existing standards/technologies and real-life bioinformatics workflows. Figure 1.2 summarises the focal point of this thesis and shows the context in which the research of this thesis is applicable.

1.5 Structure of the Thesis

The rest of the thesis as shown in Figure 1.3 is structured as follows:

Chapter 2 provides the background on provenance, provenance taxonomy, classification of workflow definition approaches and comparative literature surveys of leading WMS in the bioinformatics domain with focus on their provenance support and approaches to aggregate resources utilised in a given analysis.

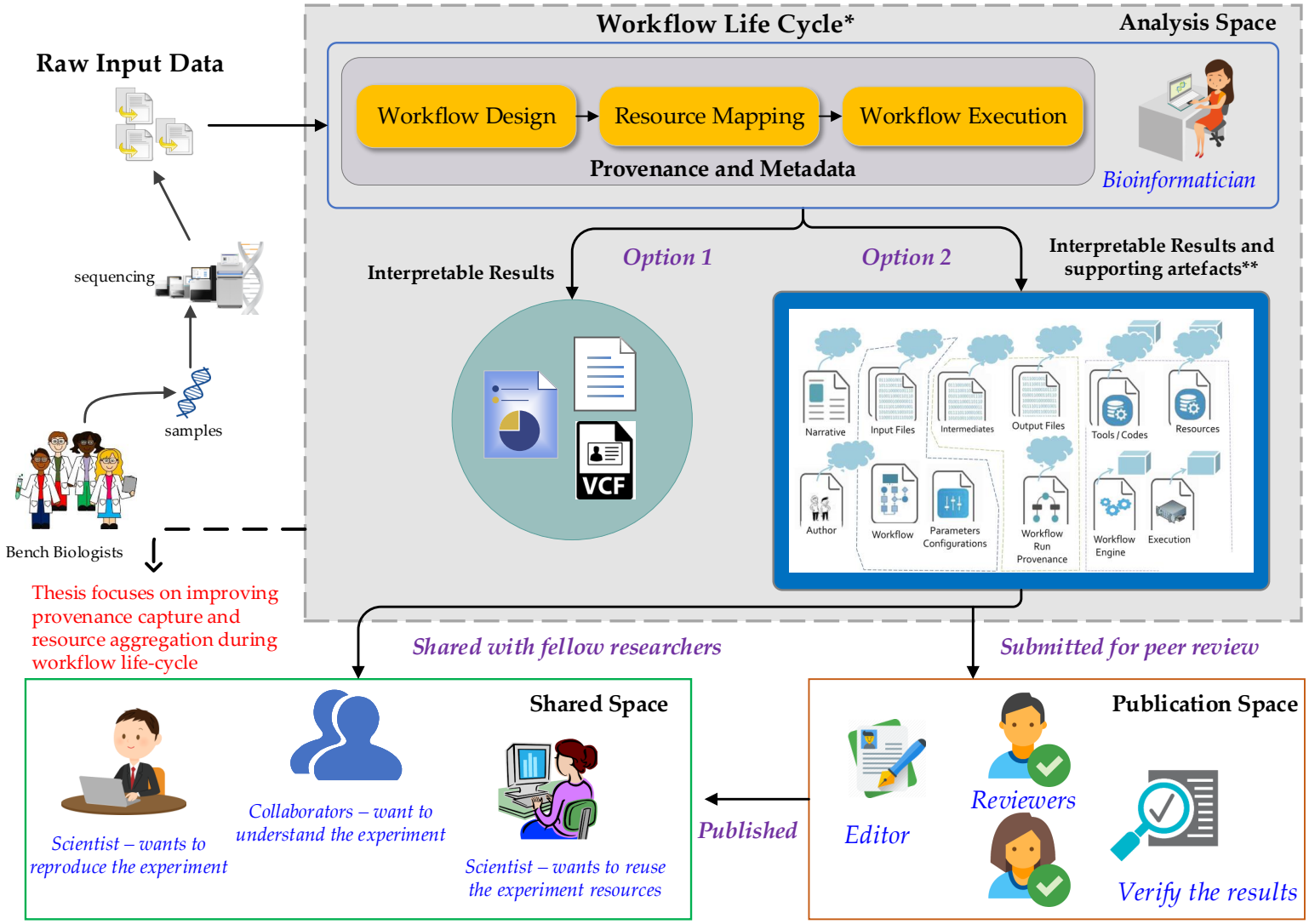

Figure 1.2: Overview of a typical bioinformatics workflow life cycle. The grey highlighted area shows the focus of this thesis. (*)The workflow life cycle is adapted from [34] and (**) the aggregation representation is adapted from [35].

Chapter 3 presents an empirical case study analysis by implementing a variant calling workflow on an exemplar systems using different workflow definition approaches. This chapter also provides details regarding the implicit assumptions made by each approach and their implications on the provenance of the workflow under investigation.

Chapter 4 focuses on the essential elements of provenance and the choices of technologies that impact on re-use and reproducibility of a study. This chapter includes detailed comparisons of the state of the art research undertaken by various groups focusing on improving workflow design, representation, sharing and publication. This chapter also presents the conceptual hierarchical provenance framework culminating from these analyses.

Chapter 5 introduces *CWLProv*, a standard format for representation of workflows, utilising existing community-driven and open-source initiatives proposed to overcome the issues with heterogeneity of workflow definition, provenance representation and resource aggregation. This chapter also includes details of the practical implementation of *CWLProv* by extending an existing implementation.

Chapter 6 focuses on illustrating *CWLProv* with respect to interoperability and reproducibility using a variety of open-source real-world workflows developed by independent groups on separate platforms. This chapter also discusses the overheads incurred as a result of provenance capture and the remaining open challenges.

Finally *Chapter 7* concludes this thesis. It includes a summary of the contributions of this thesis, limitations and potential future lines of work. It also discusses the impact of this research on the current and future studies.

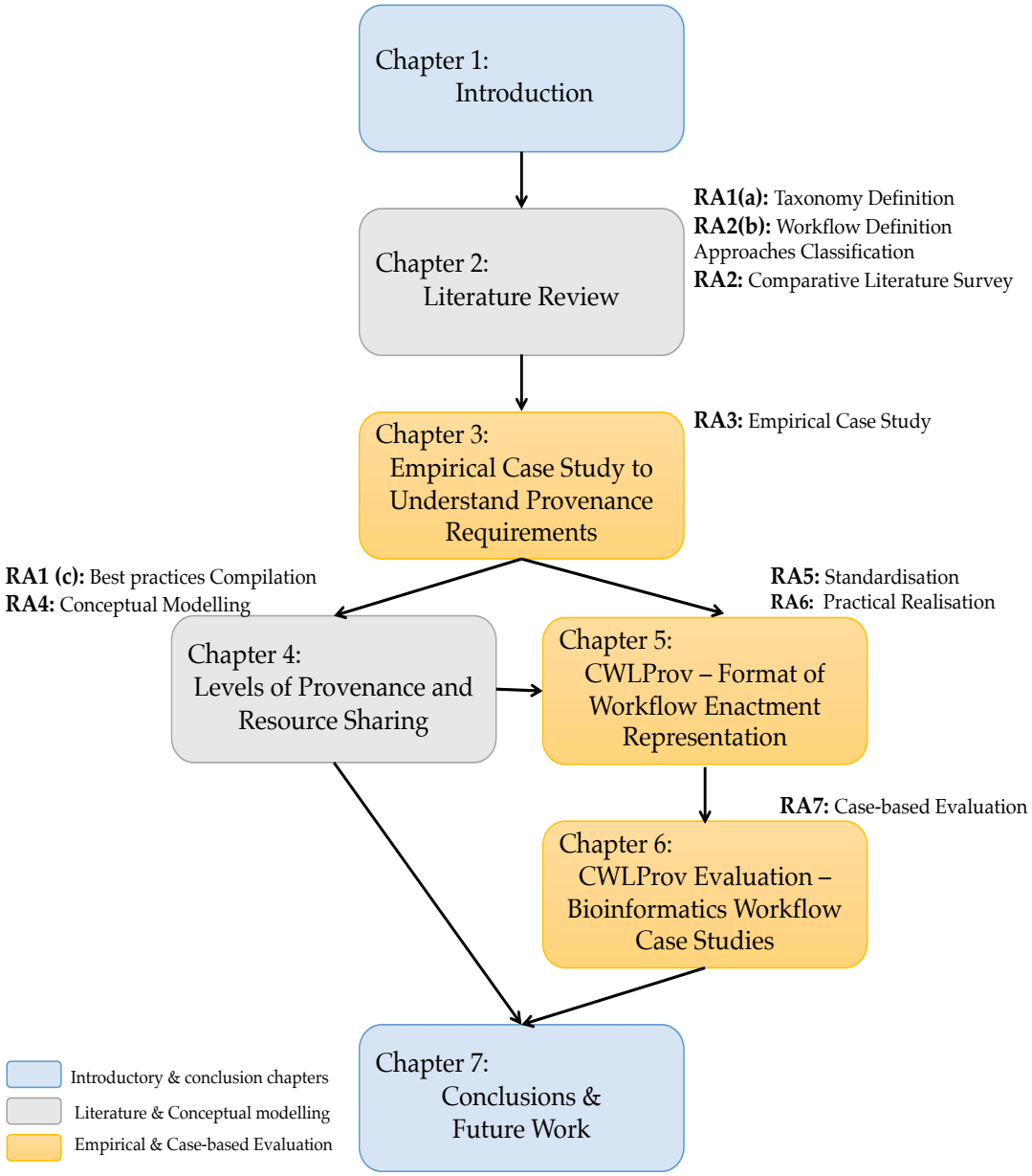

Figure 1.3: Thesis Outline mapping the Research Activities (defined in Section 1.1) that how they are performed in each chapter

CHAPTER 2

LITERATURE REVIEW

The investigation of the truth is in one way hard, in another easy. An indication of this is found in the fact that no one is able to attain the truth adequately, while, on the other hand, no one fails entirely, but everyone says something true about the nature of all things, and while individually they contribute little or nothing to the truth, by the union of all a considerable amount is amassed. –Aristotle

2.1 Introduction

In this chapter we introduce the workflow-oriented provenance taxonomy that forms the basis for the research in this thesis. The term *"provenance"* has numerous related concepts that apply or have been adopted in different domains, hence we provide clarity in what we mean by provenance with regards to this thesis. Building on this we present studies capturing and standardising provenance, and associated attempts to implement these standards and approaches by the broader scientific community. We focus especially on life sciences research and the importance of bioinformatics workflows and evaluate the leading workflow systems and their specific demands for provenance. We classify existing workflow systems into distinct, meaningful categories. The chapter concludes with a detailed evaluation of the current state of the art workflow systems.

It is noted that parts of this chapter have been published in peer reviewed academic articles listed in the Preface of this thesis.

2.2 Provenance Taxonomy

The term *Provenance* has historically originated from the arts and humanities where it was applied to trace the origin of the artwork. More recently the term has been adopted in disciplines such as geography, astronomy and computer science with various intended meanings and with sometimes overlapping definitions. Provenance has historically been referred to as lineage [36], data pedigree [37], dataset dependence [38], execution trace [39] and audit trail [40] in many studies dedicated to solving open problems in provenance tracking. The following section summarises the origin of the term and how it has been defined in different contexts.

2.2.1 Provenance Context

The term *Provenance* was predominantly used in the arts and especially in archiving domains to determine the authenticity, quality and ownership of a piece of art. The Merriam Webster dictionary defines it as:

"The history of ownership of a valued object or work of art or literature"

The Oxford dictionary has a similar definition:

"A record of ownership of a work of art or an antique, used as a guide to authenticity or quality"

These definitions depict a view of provenance and its role in establishing authenticity of a given artifact e.g. a painting, such that trust can be established and value of that artifact determined correctly. Although subsequently used in different domains, the underlying usage and rationale behind recording and keeping track of provenance is well aligned with these definitions, where the *"valued object"* could be any domain-specific artefact e.g. historical records in case of archival science, proof of origin of life in case of palaeontology or the source of big data-sets in case of astronomy. Hence, the term *provenance* is not limited to any specific domain, but rather widely adopted in present day language, serving these basic applications as well as many others described later in this chapter.

In Computer Science, the *valued object* can be any digital artefact. Provenance-oriented studies have focused on developing methods, standards, definitions and formalisation of *Computational Provenance* tracking of valued objects. The idea of formally keeping track of provenance traces back to when Bench-Capon *et al.* provided a *provenance tool* for validation and verification of knowledge-based systems [41]. Tennenhouse used the term 'sample provenance urging researchers to keep information about data samples that could aid future researchers in understanding and repeating scientific experiments [42].

Resources that do not explicitly capture provenance as a primary goal can still aid in provenance tracking. Cheney *et al.* proposed that provenance can be captured and supported implicitly through application version control, the operating system logs, the curated database entries, the file systems through to the web browser history information. Such information can aid in debugging, auditing and error handling [43]. These operations, although serving the purpose of provenance capture, were not particularly designed for it. Rather they represent a retrospective ad hoc information capture system that contribute to supporting provenance without being directly identified at the outset to support its tracking.

In past two decades, the topic of provenance tracking has gained much attention due to the increased demands for trust and authenticity of digital artefacts published on the Web [44]. Groth *et al.* explain the importance of *Provenance Systems* by calling them the *"equivalent of the scientist's logbook for in silico experimentation"*. The term *Provenance* is very broad and as such it is difficult to attain a single definition. With an increasing number of studies and research groups working on this topic, there are more than one viewpoint associated with the same term. Considering there are perspectives, it is of course critical to establish the definition of provenance used and followed in this thesis. Below we discuss few definitions presented in different studies and conclude this subsection with the one followed in this thesis.

Simmhan *et al.* [45] define *Computational Provenance* as:

"metadata that pertains to the derivation history of a data product starting from its original sources."

Zhao *et al.* [46] state that *in silico* experiments should be equipped with provenance traces. They define provenance as:

"Provenance is a kind of metadata, recording the process of biological experiments for e-Science, the purpose and results of experiments as well as annotations and notes about experiments by scientists."

Groth *et al.* [47] present a different view, defining provenance as a concept and emphasising the documentation of the *process* that produces the data artefact:

"the provenance of a piece of data is the process that led to that piece of data"

The community-sourced efforts of the World Wide Web Consortium (W3C) Provenance Incubator Group focused on understanding the importance of the

standardisation of provenance related to semantic web technologies provided recommendations in this respect and proposed the following working definition [1]:

"Provenance of a resource is a record that describes entities and processes involved in producing and delivering or otherwise influencing that resource. Provenance provides a critical foundation for assessing authenticity, enabling trust, and allowing reproducibility. Provenance assertions are a form of contextual metadata and can themselves become important records with their own provenance."

In case of *Computational Provenance* involving digital artefact generation, its formal representation and sharing over the web, the term "resource" in the above definition refers to digital artefacts. Later W3C Provenance Working Group when realising the recommendations of the W3C Provenance Incubator Group to standardise provenance representation (details in Section 2.2.4) define provenance [25] as:

"Provenance is information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness."

We follow the above stated definition in the remaining chapters of this thesis. This definition and its associated concepts e.g. entities and activities emerge from two years of disciplined community-wide effort aggregating experiences of participants from a range of application domains representing over fifty organisations. These group members contributed to the empirical and practical efforts by bringing unique requirements of their respective research domain. This definition fits well with the concepts of *Computational Provenance* with respect to database systems as well as workflows. The focus of this thesis is provenance related to

[1] https://www.w3.org/2005/Incubator/prov/wiki/What_Is_Provenance

scientific workflows, hence we provide only a brief discussion of database provenance and business workflows.

2.2.1.1 Database Provenance

Databases have been used extensively for storing scientific data for decades [48]; consequently provenance studies with respect to databases have been thoroughly investigated. As any data artefact, provenance of database entries and records also require identification of the origin, attribution and history of the data to establish trust and validity [44]. Provenance in database systems is alternatively called *data lineage* and it focuses especially on determining the source of data for a given data item [49].

Buneman *et al.* exploited the concept of *data provenance* in the context of databases and referred to it as keeping a record of the origin of a piece of data as well as the process of its creation [37]. In another study, Buneman *et al.* [50] adopted the concept of *why-provenance*, i.e. "Why is a given piece of data in the database?". An alternate view provided by [51] defines *how-provenance*, i.e. "How was a data item produced/generated?". [52] introduced a different notion, *where-provenance*, i.e. "Where did it come from?". Another equivalent term used for *where-provenance* by [49] is *input provenance* which is defined as:

"given a piece of data X, the input provenance of X is all data that contributed to X being as it is"

Provenance in databases is considered as *fine-grained* [53] and *white-box* provenance [54] because the inner working details of each database query are explicit and the computational details are readily available for further analysis or querying. The approaches to capturing and recording database provenance are gen-

erally classified as either *eager* or *lazy* [55]. Cheney *et al.* describe an *eager* approach as an *annotation* approach where queries are re-engineered to make them *provenance-enabled*, including additional annotations for provenance-rich outputs. In the *lazy* approach, the provenance is only computed when required by looking into available resources such as input data, available queries and the associated output data.

A number of reviews have addressed and summarised provenance capture and querying in the database domain, e.g. [56, 54, 55, 57]. Since the focus of this thesis is on provenance with respect to workflows, we do not discuss database provenance further.

2.2.1.2 Workflow Provenance

Before exploring what provenance means in the context of workflows, it is necessary to discuss what a computational workflow is, what the term *scientific workflow* means specifically, and, importantly, how it differs from a business workflow.

2.2.1.2.1 Business Workflows

The concept of *computational workflow* originated from the business domain where it was formally defined by the Workflow Management Coalition [58] as:

"The computerised facilitation or automation of a business process, in whole or part. Workflow is concerned with the automation of procedures where documents, information or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal"

Workflows have been extensively used in the business domain to achieve automation of business-oriented (repetitive) tasks and processes. Many studies

have been dedicated to define a standard for modelling these workflows. Mainstream approaches have included Business Process Execution Language (BPEL) [59], Yet Another Workflow Language (YAWL) [60], XML Process Definition Language (XPDL) [61], Web Services Choreography Description Language (WS-CDL) [62], and others. Indeed there are a wide variety of languages, standards and frameworks for workflows resulting in considerable heterogeneity.

2.2.1.2.2 Scientific Workflows

While the basic definition of a computational workflow for the business domain is at its core also applicable to the scientific disciplines, in order to automate the analysis of large-scale scientific data exploiting, for example, High Performance Computing (HPC) resources, a specialised term *scientific workflow* was coined and defined in [63] as:

"scientific workflows are networks of analytical steps that may involve, e.g., database access and querying steps, data analysis and mining steps, and many other steps including computationally intensive jobs on high performance cluster computers"

Altintas *et al.* [64] refer to scientific workflows as:

"the automated process that combines data and processes in a structured set of steps to implement computational solutions to a scientific problem"

Discussing data modelling challenges and approaches, Bowen [65] defined *scientific workflows* as directed graphs where each task constitutes a node connected to other nodes through a data-flow or through control-flow edges:

"A scientific workflow is a high-level description of the processes used to carry out (often complex) computational and analytical experiments. Scientific workflows are modelled as ***directed graphs*** *consisting of task nodes and data-flow or control-flow edges*

denoting execution dependencies among tasks. Each task within a scientific workflow represents a specific computational step (e.g., within a simulation study), data analysis step, or data management step"

Ludascher *et al.* list some distinctive features differentiating *business workflows* from their *scientific workflow* counterparts [66]. The most striking is the data-driven approach of *scientific workflows* - making them data-flow oriented - versus the process-oriented approach of *business workflows* which results in models based around control-flow. Moreover *business workflows* are specifically designed to achieve business goals and the outcome of such workflows is often known even before the execution. On the other hand, *scientific workflows* are often given as scientific experiments where the purpose is to use them as a tool to serve an experimental goal. Given the extensive use and emphasis on automation of computational processes using workflows to address the complexity of scientific analyses, *scientific workflows* are now considered a **valued commodity** according to [67], who states:

*"As a consequence of the lengthy, iterative design process, workflows become a **valued commodity** and a source of intellectual capital. The output of workflows or workflows themselves may be used as a basis for future research, either by the scientists who generated the data, or colleagues in a related field.[..] These workflows should be **reused**, refined over time, and **shared** with other scientists in the field. Scientific workflows must be fully **reproducible**. In order for a workflow to be reproduced, **provenance** information must be recorded that indicates where the data originated, how it was altered, and which components and what parameter settings were used. This will allow other scientists to re-conduct the experiment, confirming the results"*

Deelman *et al.* [34] postulate that workflows have enabled the automation of data building processes and their associated provenance. The authors classified the lifecycle of a typical workflow into four stages as shown in Figure 2.1.

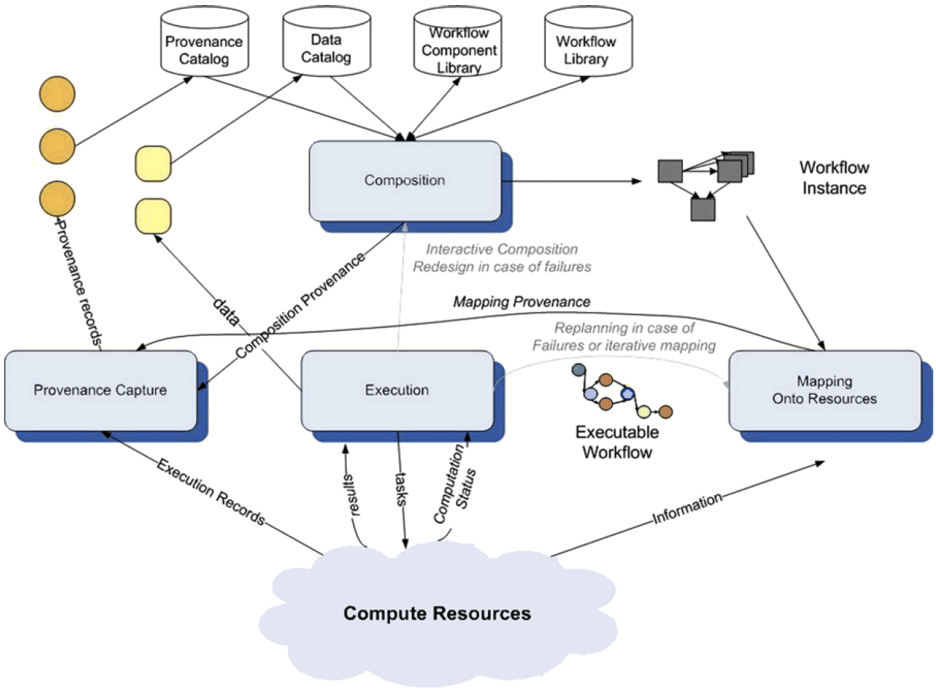

Figure 2.1: Workflow lifecycle comprised of four stages: composition, mapping, execution and provenance capture where the last stage can overlap with the previous stages

We consider the aspects of a typical workflow lifecycle.

- **Composition, Representation and Data Model:** The first stage is to create a workflow either from scratch or by re-using existing workflows obtained from collaborators or open access repositories. The users can use various composition methods, e.g. workflow language representations, higher level scripting languages or Graphical User Interfaces (GUI). The representation of the workflow is most commonly given as a Directed Cyclic Graph (DCG) or as a Directed Acyclic Graph (DAG). It is the composition stage when workflow authors define whether the workflow is data-driven (the task dependencies and connections are managed by the flow of data from one task to the other) or control-driven (flow of control such that execution of a given task depends on completion of the preceding task).

- **Mapping:** This stage allows users to either perform resource mapping for a workflow execution directly or use a system which is designed for workflow execution. The composition should ideally result in an *abstract workflow* that can be mapped to any target execution environment. Mapping results in resource bindings to the available resources such as tools, compute, storage, external services, etc. Deelman *et al.* discuss partitioning the workflow into "sub-workflows" before mapping occurs to improve scheduling in case of distributed computing and if possible, running the tasks in parallel.

- **Execution:** Once the workflow has been mapped to the available resources, it can be executed (often referred to as "enacted") using an execution engine or any enactment system. An important aspect at this stage is utilising fault tolerance which can be at the Operating System (OS) level, at the application/tool level or in the workflow itself.

- **Metadata and Provenance:** The lifecycle of a scientific workflow comes to fruition with the collection of metadata and provenance information. Deelman *et al.* argue that if a data artefact was generated by a workflow, then there must be a way to store the history of the creation process. They used three key terms for provenance: *Data Provenance*, *Design Provenance* and *Workflow Execution Provenance*. The authors also believe that although this stage is categorised separately, it can overlap the first three stages including collecting data, metadata and provenance information throughout the workflow lifecycle.

A similar classification of the scientific workflow lifecycle is put forward by Ludascher *et al.* [66]. The authors state that each workflow is directed towards achieving an experimental goal or testing a scientific hypothesis. They identify four key stages: *workflow design and composition*, *workflow resource planning*, *workflow execution*, *workflow execution analysis* and *workflow and result sharing*. The *work-*

flow design and composition is equivalent to the *Composition, Representation and Data Model* stage presented by Deelman *et al.*. The *workflow resource planning* is well aligned with the *Mapping* and *workflow execution* with the *Execution* stage. The fourth stage *workflow execution analysis* focuses on analysis, debugging and evaluation of data products, execution traces and data dependencies on the workflow. The last stage in this case is more analysis-oriented and uses metadata and provenance information, unlike the fourth stage by Deelman *et al.* where the focus is only on collection of such details but not the post-analysis part of the workflow cycle. The fifth stage *workflow and method sharing* refers to publishing and making the methods and data used in an experiment available, ideally via a shared repository.

Cuevas-Vicenttín *et al.* [3] also assert that *scientific workflow* design and management has become an essential part of many computationally driven data-intensive analyses, enabling ***A**utomation*, ***S**caling*, ***A**daptation* and ***P**rovenance support* **(ASAP)**. The authors identify that the goals of scientific workflows are in facilitating easy workflow design, re-use, scalable execution and ultimately *reproducible science* explicitly through provenance support.

Davidson at al. [68] illustrate the importance of provenance information of a workflow by discussing its uses. The authors put forward the idea that not only can provenance be used for data interpretation, but also for trouble-shooting and optimising efficiency. For example, multiple re-runs of a workflow for evaluation and experimental purposes can typically require altering the parameter space for any tool. In this case, the provenance can improve efficiency by providing information about a specific *checkpoint* in a workflow run and re-running the workflow from that point. Alper *et al.* [69] indicate that, using provenance traces of workflows when generating a dataset, it is possible to automatically satisfy the requirements of provenance and context of metadata provision along with the

published dataset. This was also identified by the Open Archival Information System (OAIS) Reference Model [70].

2.2.1.2.3 Provenance and Scientific Workflows

It is evident from the above studies that provenance capture is considered an essential feature of scientific workflows. The term *computational provenance* has been re-defined, categorised, and updated to cater the specialised needs and requirements of workflow-centric research.

Simmhan *et al.* [45] defined workflow provenance with respect to scientific data analysis in the context of Service-oriented Architecture (SOA) as:

"metadata describing the workflows execution and associated service invocations. Workflow provenance typically includes information on the services invoked, the order and time of invocations, parameters passed and returned, and faults that occurred, among others."

The *services* in this definition can be replaced by any pre-built local computational tool used to achieve a task in the workflow if a SOA is not used for workflow execution. Another closely related definition of scientific workflow is put forward by Boher in his article [71]:

"It refers to the records of the history of derivation of a given final output of a workflow which is typically used for complex processing tasks."

Moreau [44] compares the provenance with a log book in the survey article:

"provenance is regarded as the equivalent of a logbook, capturing all the steps that were involved in the actual derivation of a result, and which could be used to replay the execution that led to that result so as to validate it."

These general definitions of workflow provenance do not differentiate between the specific aspects of provenance associated with the scientific workflow domain such as provenance of the workflow specification, workflow enactment and workflow evolution which are discussed in the next section and revisited in this thesis later.

A key distinction between database and workflow provenance is the expected granularity of the provenance information. As discussed earlier, database provenance is often referred to as *fine-grained* or *white-box* because of the extensive details available about the computations. Workflow provenance however is typically called *black-box* as the individual tasks, modules or services typically provide *coarse-grained* information of the underlying tool used for processing. As each workflow step can be implemented using different resources such as invoking shell scripts, third-party web services or another workflow, the granularity of provenance for each step could vary notably. Therefore, an intermediate notion of workflow provenance granularity called *grey-box* is proposed by Bowers & Ludäscher [72]. *Grey-box* denotes the varying degree of granularity between different processes of a scientific workflow depending on the underlying tool resulting in partial transparency.

Tan [54] explains the nature of workflow provenance with the help of a sample workflow adapted from [73] as shown in Figure 2.2. We consider DAG representations of workflows in this thesis, as they are the predominant approach used in bioinformatics. This sample workflow comprises four steps. *S1* downloads a set of DNA sequences from GenBank (an online repository). *S2* takes these downloaded sequences and runs an alignment tool/program to generate multiple sequence alignment. This step can use any appropriate underlying tool for this purpose and the exact algorithm used to generate this alignment is often implicit, resulting in a *black-box*. *S3* involves human intervention where the

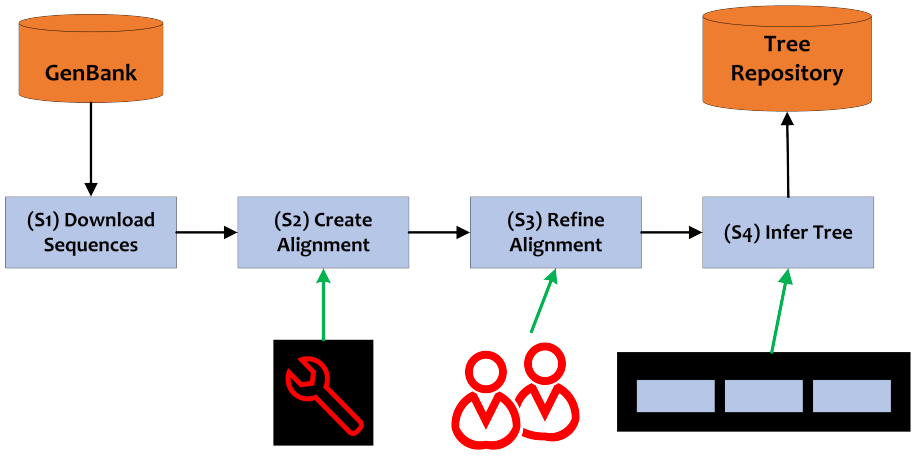

Figure 2.2: An example workflow adapted from [54] to show the coarse-grained nature of workflow provenance

generated alignments are examined and altered if necessary. This is a separate issue (not discussed in this section) as a workflow not being fully automated results in incomplete documentation of the process followed resulting in missing information. *S4* accepts the final results of alignment (after S3) and generates a phylogenetic tree which is saved in a repository. *S4* is a multi-step process [73] and hence is an abstract representation where the details are hidden and result in a coarse-grained view of provenance similar to S2.

2.2.2 Workflow Provenance Classes

Previous studies have categorised the provenance related to workflows based on application demands, required artefacts and/or the nature of the scientific experiment output. This has led to various studies focusing on different aspects of provenance, resulting in ad hoc solutions tackling one or more specific classes of provenance. This section includes definitions of these provenance classes and discussion about details of the artefacts required to be captured for each class.

2.2.2.1 Prospective Provenance

Zhao *et al.* [74] view prospective provenance as a recipe encompassing *"all the aspects of the procedure or workflow used to create a data object"*. Prospective provenance refers to the *recipes* used to enact a computational task, e.g. the workflow [75]. It is an abstract representation of the steps that are necessary to create a particular research output/data artefact. Any resource that declares the way to enact a given step in a workflow or the workflow as a whole is considered part of prospective provenance. A workflow specification can be enacted multiple times using same or different datasets by the same or different users. Ludäscher [53] identified that a well-defined workflow was itself a source of prospective provenance. If a workflow is well-documented such that users can infer from the workflow graph the general method for the result production, little effort is required to transform this information into a standardised format representing the prospective provenance record. An example *Data-flow* workflow graph is shown in Figure 2.3 depicting the dependencies between the steps based on the data artefacts produced and utilised. Prospective provenance is typically collected during the *"Composition"* stage of the workflow life-cycle.

In general, the *recipe* or prospective provenance in the case of a workflow can include any resource which helps users to understand the requirements of the workflow as well as to subsequently support its enactment. The notion of methods as first class objects [77] to increase transparency of the methods and credibility of results can be adopted and supported if the prospective provenance is enriched with necessary resources. Such resources include workflow specifications, an abstract workflow representation (e.g. the graph as shown in Figure 2.3), example datasets, annotations of input and output data, required/recommended compute and storage requirements and a description of the actual purpose of the workflow.

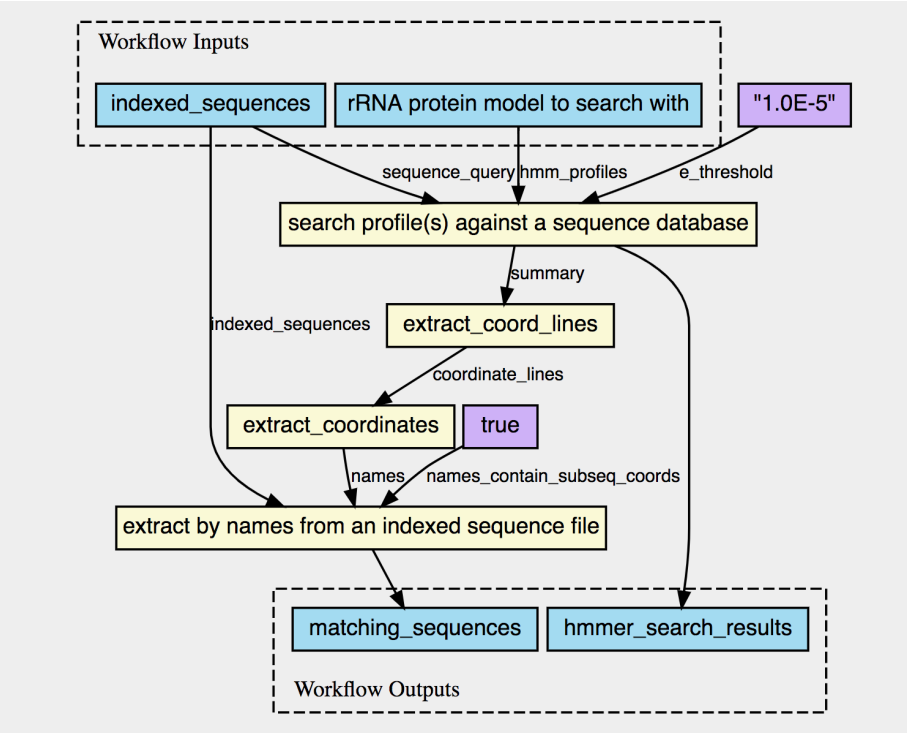

Figure 2.3: Example data-flow workflow defined in the Common Workflow Language [31] depicting prospective provenance. The diagram is obtained from the *CWL viewer* [76]. The yellow nodes denote the steps/processes and the edges denote the order of their execution and hence data dependencies between jobs to be executed.

2.2.2.2 Retrospective Provenance

The derivation history of a given data artefact can be classified as retrospective provenance. In the case when the method applied to derive/produce a data artefact is a workflow, retrospective provenance refers to the details of the workflow enactment, the run-time environment and the resources utilised to produce a given data artefact [74]. Lim *et al.* [78] define retrospective provenance as:

"Retrospective provenance models past workflow execution and data derivation information, i.e., which tasks were performed and how data artefacts were derived"

Missier *et al.* [79] consider retrospective provenance as an annotated Directed

Acyclic Graph (DAG) recording the history of data consumption and production. As it records the history, it should be immutable. Nodes could either be pieces of data if the provenance is recorded for a data-flow oriented workflow, or an activity if the workflow is control-flow oriented. The edges of the DAG are the relationships between activities, or the data showing the link between different steps or stages of a given workflow.

The derivation history improves the understandability of an experiment [53] and if accompanied with the artefacts utilised in the workflow enactment, it can help users re-use shared methods for verification of results or for another analysis with different data [49]. These artefacts can include, amongst other things: the workflow specifications used for the enactment; information about input and output data for each step; intermediate results generated; software names and versions; Operating System (OS) dependencies; domain specific annotations, utilised compute and storage resources. These factors should be documented and stored as provenance information during the *"Execution"* phase of the workflow lifecycle. There are various provenance models designed to represent retrospective as well as prospective provenance, which are outlined in the next section.

In summary, retrospective provenance addresses questions like *"Who enacted the workflow?"*, *"what was used to create a given data artefact?"*, *"when was the workflow and its processes enacted?"*, *"Where was the workflow enacted?"*. These general questions cover roughly all the artefacts required to be documented for retrospective provenance. Lim *et al.* [78] uses Entity Relationship notation to show typical associations of prospective provenance and retrospective provenance as shown in Figure 2.4.

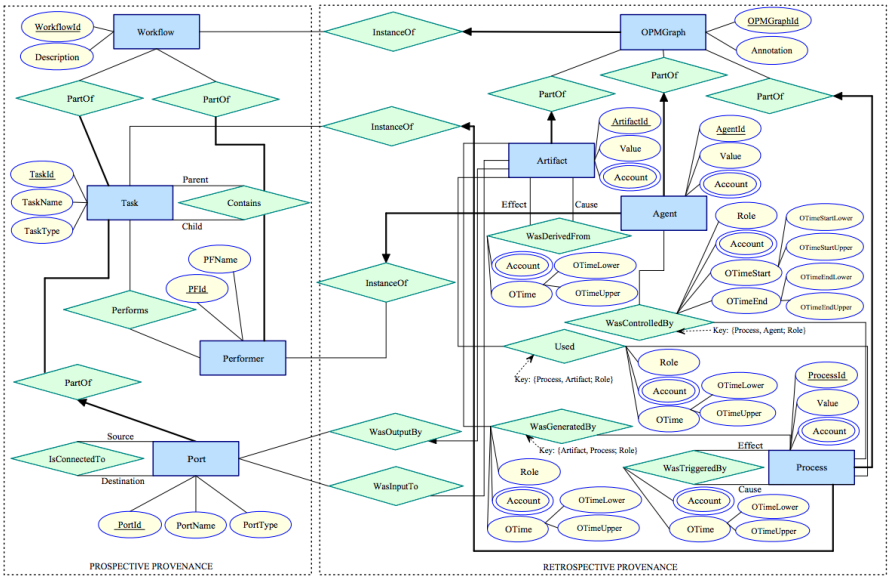

Figure 2.4: Each workflow can be run more than once resulting in workflow instances (also referred as runs or enactments) performing all of the tasks the workflow is designed for. The abstract representation on the left hand side is a recipe which is followed by the workflow instance that is run on the right hand side.

2.2.2.3 Workflow Evolution Provenance

The term "workflow evolution" was originally defined by Casati *et al.* [80] in the business domain as "changing existing workflows while they are operational".

For scientific workflows, workflow evolution provenance refers to tracking and capturing changes in workflow specifications, parameter setting [66], change in the underlying software for a workflow step, or altering (adding/removing) a step. Scientific workflows are predominately exploratory in nature such that changes in parameter space and even tools used are very common and often used to investigate performance of different settings and analyse their effect on results. Therefore, automated workflow evolution tracking is highly relevant to the scientific workflow domain.

Friere & Silva [81] emphasised the need for creating strong links using persis-

tent identifiers between workflow results and the exact version of the workflow used to generate those results. Withana *et al.* [82] argue that workflow evolution benefits research by keeping track of changes in workflows and associated parameters over time, with the opportunity to revert back to default settings after evaluating results. The authors also found that given information of the exact version of the workflow, result comparison and attribution are easily achieved. They classified the changes into two types: *direct evolution* and *contributions*. *Direct evolution* refers to changes in the flow and arrangement of the processes; changes in components within the workflow; and changes in configuration/parameter settings. *Contributions* on the other hand, refer to tracking information such as *where this workflow came from?*. It focuses on documenting and tracking the lineage of the workflow. This classification differentiates between changes made by the researcher's direct involvement and changes made by someone else.

This thesis is primarily concerned with the notion of *data-flow* workflows, and the focus is on retrospective, *process-oriented* provenance [45]. In *process-oriented* provenance, the provenance of the processes (workflow and its intermediary steps) is tracked and used to determine the lineage of a given data artefact. It is to be noted that *workflow evolution* is also defined and classified in this section for completeness.

Scientific workflows provide effective computational tools as described in Section 2.2.1.2. Ludäscher *et al.* [66] summarise the advantages of scientific workflows with one directly associated to provenance. They discuss that scientific workflows have an edge over ad hoc approaches, e.g. scripts as they require explicit documentation of the computational process making it subsequently possible to track provenance. Complete documentation of the process, including retrospective provenance, leads to improved understandability, effective sharing and reproducibility of experimental processes and results. In the following section,

applications of workflow provenance are discussed to highlight the importance of provenance documentation and sharing for use in different scenarios.

2.2.3 Provenance Applications

The purpose and utility of provenance information has been discussed and categorised in various studies such as [83], [84] and [85]. These studies have covered the application of provenance in detail, emphasising the need for complete capture of provenance to benefit from the automated analysis achieved through workflows. In this section, relevant applications from literature are summarised and explained in the context of scientific workflows.

2.2.3.1 Understandability

In the context of scientific workflows, all three classes of provenance are important for better understanding of a given analysis. Prospective provenance helps researchers inspect the requirements of the workflow to be run. Information about the underlying goal (hypothesis) of the workflow as part of prospective provenance can be used to answer questions related to the purpose of the workflow and the circumstances in which it should be used. The details of the working methods can also be inferred from prospective provenance. Retrospective provenance provides insights about the experimental settings for a specific workflow enactment. The user can inspect retrospective provenance to understand the effect of a given set of parameters on the results which can help in planning subsequent experiments, e.g. for result comparison. Domain-specific metadata about inputs and outputs also helps to understand the context and expected file formats required to run a workflow [86].

This application of provenance is important in various places, e.g. sharing analyses with collaborators, submitting workflows for publications and most importantly for the researchers themselves when revisiting the analysis in the future. Retrospective provenance can be viewed as a structured graph; however one challenge is the increased size of more complex workflows. This results in a cluttered view of provenance and reduces the utility of the provenance information. Efforts to increase understandability by abstraction and summarization of the captured provenance have been considered [69, 87, 88].

2.2.3.2 Attribution

Keeping a record of attribution details as components of provenance has been emphasised by Goble [83], Cheney *et al.* [55], Garijo *et al.* [89] and various other studies. According to Murphy [90]:

'Attribution is the hallmark of scholarship: statements and works are attributed so that credit can be assigned and provenance and responsibility can be determined."

In the context of workflows, details of both the workflow authors and the workflow users who may adapt the workflow for their own analysis, should be documented. Storing attribution details about workflow authors ensures proper accreditation which is often ignored if this key information is missing. If workflows and their related resources are considered first class data objects [91] like other software, accreditation should be transparent. It establishes the ownership of the workflow and data used in an analysis. It also helps the creators evaluate usage of their resources [45]. In addition to accreditation, attribution details can also be used to assign responsibility in the case of erroneous results. Complete provenance traces record such information and can be used to avoid any illegitimate claim of attribution. With increasing open-access resources and com-

putational experiments, it is now crucial to establish such references for proper attribution ideally supported by digital signatures [92].

2.2.3.3 Reproducibility

Provenance of scientific workflows has been considered a key aspect of computational reproducibility of analysis in various studies [64, 7, 93, 94, 95, 79, 96]. Reproducibility of computational experiments using workflows often requires replication of the precise software environment used in the original analysis. Detailed provenance information including details of the software, data and resources used for the workflow enactment aids in the reusability of the methods. Provenance tracking and reproducibility are closely associated since provenance traces can help make any research process auditable and results verifiable [97].

Once a computational experiment involving a workflow is shared, it is reasonable to expect that experts from the same domain will re-enact the workflow, either to reproduce the results to confirm the findings or to re-use methods for new/similar datasets for comparative studies. Analysing rich retrospective provenance traces containing details of the operations performed during a workflow enactment to derive a data product can help under these circumstances. The utility of provenance information for reproducing an analysis depends on the availability of methods and data alongside the provenance trace [84].

2.2.3.4 Authenticity, Transparency, Trustworthiness (ATT)

The important question, *Can we trust this method?* or *Is this data reliable?* can be answered if the original and lineage history of a data artefact is available. Information and details about methods used in an experiment to produce a data

artefact are crucial to establish credibility of scientific results [77]. In the case of workflows, this information can be inferred from both prospective and retrospective provenance of the workflow. The concept of using provenance information to support trust in a given artefact is as old as the provenance notion in the field of arts. This concept also holds for digital artefacts where the provenance information plays an important role in trust judgements about the shared/published resources ultimately supporting *Trust on the Web* [98].

Groth *et al.* [84] identifies that trust and authenticity often link back to the attribution details provided. They emphasise that *"Without provenance, information is hard to understand, integrate, and trust"*. Gamble & Goble [99] describe three *concerns* a researcher would have when deciding to use any data available on the web. One of these concerns was *Trust* of the data based on its provenance. In the case of using workflows as methods in a computational experiment, this information can be tracked during workflow enactment and support for retrospective provenance. In addition, well-defined workflow specifications including details of attribution also aid in establishing trust of the methods used. Authenticity of scientific results, data sets and claims can only be established if enough documentation about the methods utilised to produce these results is also shared and published with the associated results and methods [100].

2.2.3.5 Quality Assurance

With the profusion of scientific data disseminated on the Web, researchers also need mechanisms to access and evaluate the quality of these data artefacts [99]. If complete documentation of the data derivation in the form of provenance is available, this information can be used to determine the source of data, the attribution details and the methodology used to derive the data artefact. Retrospec-

tive provenance in combination with execution and domain data is also helpful in improving the quality of the methods used in a given analysis by assessing and debugging the workflow executions [101]. The following quality dimensions proposed by Gamble & Goble can be adapted for both data and process quality assessment of scientific workflows:

- *Quality Dimension* evaluates the quality of the data in comparison to the established standard data formats, information models and vocabularies. When sharing an artefact (data or workflow), the researchers should adhere to pre-defined formats and standards for their representation.
- *Trust Dimension* evaluates if a given data artefact is accompanied with sufficient provenance details, e.g. derivation history, persistent identifiers, information about parent/source data and attribution.
- *Utility Dimensions* evaluates if the given data artefact or method meets the requirements to be used or re-used in a particular experiment. This dimension is directed to evaluate the relevance of the artefact (method or data). This dimension can be satisfied if well-annotated data and workflows are shared with information about the purpose of the analysis and the hypothesis tested.

This section described the mainstream applications of provenance in the context of scientific workflows. To fully utilise the captured provenance information, a common standard for provenance representation is recommended that can be unambiguously used, queried, exchanged and understood to avoid heterogeneity issues. To address this, many community-driven and system-independent models of provenance have emerged in the last decade. The next section includes state of the art standards and models of provenance representation in the specific context of scientific workflows.

2.2.4 Provenance Standards and Models

In order to understand provenance information, there needs to be a structured representation of provenance based on a commonly agreed upon vocabulary such that the information can be exchanged and inferred without depending on one particular software system. The efforts towards such a common data model for provenance representation informally started in 2006 during the International Provenance and Annotation Workshop[2] (IPAW06). Participants of the workshop discussed the issues related to data provenance, data annotation, process documentation and data derivation. They concluded that the scientific community required a better understanding of existing provenance approaches and the similarities and differences of their associated representations [102]. To address this, a series of *Provenance Challenges* were initiated, the first two challenges leading to a agreement over a core representation. Various teams (16 in the first and 13 in the second challenge) participated in the first two challenges to craft and present an interoperable provenance data model - the *Open Provenance Model (OPM)* in 2007 [24].

The third provenance challenge[3] focused on interoperability of the proposed model and resulted in a governance model for the standard together with improvements in the specification [103]. The fourth and final challenge of this series also had interoperability as the main theme, and was terminated early to merge efforts with events at the World Wide Web Consortium (W3C) Incubator on Provenance [104]. The Provenance Working Group took this initiative onward to support the representation, publication and use of provenance information of data, documents and other resources over the Web. The goal was to specify a core extensible provenance language which could be adapted by any system and do-

[2]https://dblp.org/db/conf/ipaw/ipaw2006
[3]https://openprovenance.org/provenance-challenge/ThirdProvenanceChallenge.html

main, thereby promoting a homogeneous solution to provenance representation and exchange.

The Provenance Working Group formally defined provenance (included in Section 2.2.1) and presented *PROV-PRIMER* for provenance interchange. In addition, based on the recommendations of the Provenance Incubator Group [105] and building upon OPM, a family of *PROV* specifications was defined [25] for provenance representation. These were used to formally document the associated computational processes and support the aforementioned provenance recommendations.

Both of these models were based on various existing vocabularies such as Dublin Core (DC) [106], Web Ontology Language (OWL) [107] and Provenir ontology [108]. PROV emerged as an interoperable and extensible standard which could be extended for specific use cases. As the focus of this thesis is on interoperable and domain-independent solutions towards provenance representation for scientific workflows, details about specialised models targeting specific domain requirements, e.g. Proof Markup Language [109], Provenir Ontology [108] or VOID vocabulary [110] are not discussed. For the purpose of this thesis, two models have significance: *PROV* and ontologies from Wf4Ever project[4].

2.2.4.1 Open Provenance Model (OPM)

The OPM was the first model designed to represent retrospective provenance. According to the specification, it is a *"model of artefacts in the past, explaining how they were derived"* as well as a model which focuses on *"processes that occurred in the past i.e. they have already completed their execution"*. It formally defines three key terms, *Artifact* represented by ellipses, *Process* represented by rectangles and

[4]http://www.wf4ever-project.org

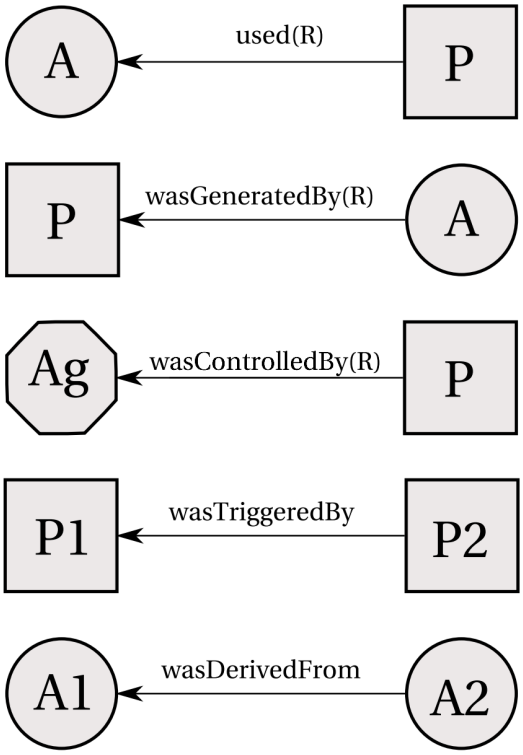

Figure 2.5: Artifacts represented by ellipses and processes as rectangles can be linked to each other other and edges are labelled based on the nature of the dependency [103]

Agent represented by octagons. *Artifact* is defined as a physical or digital object in its immutable state. *Process* is defined as a series of actions on an artifact or caused by artifacts resulting in new artifacts. Any catalyst of a process which may have enabled, facilitated or affected a process is referred to as an *Agent*.

The three key nodes can be linked to each other with edges represented as arcs between nodes. These arcs are used to denote one of five causal dependencies between nodes including: *used, wasGeneratedBy, wasControlledBy, wasTriggeredBy* and *wasDerivedFrom* as shown in Figure 2.5.

As the OPM standard is defined to represent provenance of any *"thing"*, it was deliberately kept extensible to incorporate aspects of application domains. A

noteworthy effort to extend OPM for scientific workflows and their provenance is an interoperable DataONE-OPM (D-OPM) [111]. This provides a model designed for the Data Observation Network for Earth (DataONE) project[5]. This model captures information covering many fundamental aspects of DataOne scientific workflows such as workflow execution contributing to represent retrospective provenance, workflow structure addressing prospective provenance, workflow evolution and data structures used to represent data processed by the workflow. Garijo & Gil [112] also demonstrate an extension to OPM (OPMW), designed for the Wings platform [113] to capture retrospective and prospective provenance of scientific workflows.

2.2.4.2 W3C PROV

The W3C recommended the generic standard PROV Data Model (PROV-DM) (Figure 2.6). This is predominantly adapted and extended by the scientific community working in the workflow domain and across different disciplines. As explained in the specifications, PROV-DM accommodates different provenance perspectives such as *agent-centric*, *object-centric* and *process-centric* provenance. As noted, for the purpose of this thesis, the focus is on the *process-oriented* view of provenance.

2.2.4.2.1 Core Elements

Like OPM, PROV-DM also has three key elements: an ***Entity***, ***Agent*** and ***Activity***. These elements are explained below.

- ***Entity*** is defined as any physical, digital or conceptual artifact such as a document, an image, a chart or a dataset. In the case of workflows, an entity

[5]https://www.dataone.org/

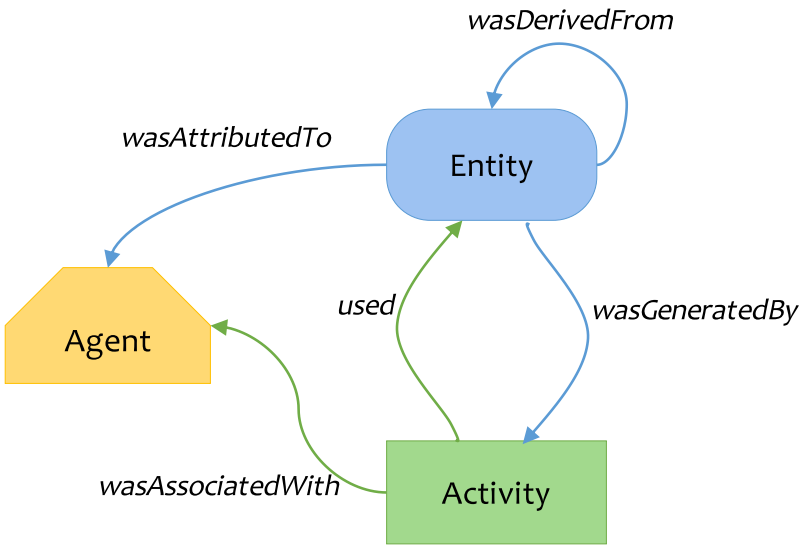

Figure 2.6: Core concepts of the PROV Data Model. Adapted from W3C PROV Model Primer [114].

can refer to input datasets, intermediate outputs, and/or final outputs. In addition, workflow specification documents can also be treated as entities as they contribute towards documenting prospective provenance.

- ***Activity*** is a process or series of processes that take entities as inputs and produce new entities. A workflow enactment is classified as an activity and also each individual step/process is referred to as an activity.

- ***Agent*** is defined as a person, software, an engine, a system, an organisation or other entities that are assigned responsibility for a given activity or an entity.

2.2.4.2.2 PROV-DM Relations

The core elements are linked to each other with defined relationships as shown in Figure 2.6. An entity can be generated as a result of an activity and this relationship is denoted as ***wasGeneratedBy***. For example, the relationship between a workflow enactment (an activity) and its output (an entity) can be represented

using *wasGeneratedBy*. Similarly, an activity (e.g. workflow enactment) can use multiple entities (e.g. input data for a workflow enactment) as inputs to generate an entity (e.g. outputs after a workflow enactment). This relationship is represented as ***used*** where the entity is *used* by a given activity. When an entity depends on another entity with respect to content or existence, the former is *derived* from the latter and this relationship is represented as ***wasDerivedFrom***. In the case of workflows, the relationship between input data to the workflow or a given workflow step, and the resultant output data produced as a result of a given step or complete workflow execution is represented using this notion.

An activity can be associated with an agent (using ***wasAssociatedWith***) and an entity can be attributed to an agent (using ***wasAttributedTo***). Another property associated with agents can be used to assign additional responsibilities by recording whether the agent acted on behalf of another agent. This is represented by ***actedOnBehalfOf***. An example of this property is given by Closa *et al.* [115] where they linked the developer of an algorithm with the executor of the process. In workflow-centric provenance, the same approach can be utilised to represent a link between the workflow author and the workflow executor (either a person or a platform). For *control-flow* workflows, another relevant property ***wasInformedBy*** connects activities and their dependencies. For example, if an activity, e.g. a workflow step "is informed by" another activity e.g. an earlier workflow step, the former will not start until the latter has completed its execution.

2.2.4.2.3 PROV Serialisations

PROV-DM is the core of the PROV standard used for defining a common vocabulary to describe provenance. It is represented by different serialisations including **PROV-O**, **PROV-XML** and **PROV-N**. The same namespace[6] for **"prov"** is used for

[6]http://www.w3.org/ns/prov#

all of these serialisations. PROV-XML serialises the PROV-DM to XML whereas PROV-O defines an OWL ontology for PROV-DM intended for Linked Data. It is to be noted that not all serialisations are *W3C Recommendations*, e.g. PROV-XML is a *Working Group Note*[7]. In this thesis, PROV-N has been adapted and detailed in *Chapter 5* as the recommended format for provenance.

2.2.4.2.4 PROV Extensions

As stated previously, PROV can be extended to fulfil requirements of different scientific domains and research areas. Several notable examples are as follows.

- ***P-Plan:*** Similar to OPM, PROV has been designed to capture provenance of events that have already happened. In the case of workflows, the model can represent retrospective provenance; however it lacks structures to formally describe prospective provenance. Garijo & Gil present *P-Plan* [116], building upon PROV's term *"prov:Plan* and their previous work [112] to facilitate the capture of workflow plans (specifications) and their relationships with workflow enactments.

- ***D-Prov:*** Based on D-OPM [111], Missier *et al.* also extended PROV specifying D-Prov to support prospective provenance by formally documenting details of the workflow specification [117].

- ***ProvONE:*** the *DataONE* project extended D-Prov resulting in a new ProvONE OWL-based specification [118]. This included more entity types such as *provone:Document*, *provone:Data* and *provone:Visualization*, as well as specific agents such as *provone:User* etc. These extensions were specific to the DataONE scenarios but could be adapted for other projects.

[7] https://www.w3.org/2005/10/Process-20051014/tr#WGNote

- ***PROV-Wf:*** Costa *et al.* [119] propose another extension of PROV targeting retrospective provenance of scientific workflows, with specific focus on those enacted in HPC environments. In this approach, the results are tracked for parallel executions of workflows and provenance data stored for further analysis.

A PROV extension that is utilised for the workflow-centric retrospective provenance in this thesis is proposed by the Wf4Ever project in the context of Research Objects (explained in Section **2.2.7.2**). This is used to describe retrospective provenance using *The Workflow Provenance Ontology* (***"wfprov"***) and prospective provenance using *The Workflow Description Ontology* (***"wfdesc"***) as explained in the next section.

2.2.4.3 Wf4Ever Provenance Ontologies

Belhajjame *et al.* [29] proposed a suite of ontologies extending existing well-known ontologies including the W3C PROV ontology (PROV-O). Based on their previous work and related literature, they presented five requirements determining the type of data and metadata that should accompany any workflow study to promote preservation of workflow-centric research. These requirements included well-described and annotated workflows and provenance traces of the workflow enactments. In response to these requirements, the authors proposed a collection of ontologies, with two specifically used to address retrospective and prospective provenance of workflows.

2.2.4.3.1 *wfdesc*

As the PROV Data Model specifically deals with the processes that have happened in the past, i.e. can only be used for retrospective provenance, the Wf4Ever

project builds upon the concept of *prov:Plan* and proposes structures as *wfdesc* to describe the prospective provenance of a given workflow. A workflow specification is described using *"wfdesc:Workflow"* which can contain two or more steps represented as *"wfdesc:Process"* with the link between the former and latter described as *"wfdesc:hasSubProcess"*. Other classes include *wfdesc:Artifact*, *wfdesc:Input*, *wfdesc:Output* and *DataLink*.

2.2.4.3.2 *wfprov*

wfprov[8] is an extension of PROV-O used to describe the retrospective provenance of workflow enactment. Four main classes of *wfprov* (on which most of the terms are built) are "artifact", "Process Run", "Workflow Engine" and "Workflow Run". A Workflow enactment (Activity) is represented using *"wfprov:WorkflowRun"* and a step (Activity) represented as *"wfprov:ProcessRun"*. The data items (Entity) are described as *"wfprov:Artifact"* and a given data item can be used as an input or produced as an output represented as *"wfprov:usedInput"* or *"wfprov:wasOutputFrom"* respectively. Another specialised extension to PROV is to include a description of the workflow engine through *"wfprov:WorkflowEngine"*, where an Agent is responsible for a given workflow enactment.

Both of these ontologies are explored in *Chapter 5* to document elements of prospective and (pre-dominantly) retrospective provenance of workflow enactment.

2.2.5 Provenance Capture Mechanisms

A uniform mechanism to capture provenance information across all computational analyses is unrealistic, given the variety and heterogeneity of approaches

[8] https://w3id.org/ro/2016-01-28/wfdesc

used to carry out these analyses (later discussed in Section 2.3). Therefore, provenance capture is typically viewed as a system-specific activity preferably without any/many infrastructural changes in the architecture of the underlying script or system. This results in specialised solutions to provenance capture tied to a single platform. The following is a classification of the provenance capture mechanisms typically used in scientific workflows.

2.2.5.1 Script-based Capture

Scripting languages such as Python, Perl and R have been widely adopted to analyse scientific datasets by defining data processing steps tied together using data or control flow dependencies. Mattoso *et al.* [120] decompose the life-cycle of a script-based workflow into three stages: composition, execution and analysis. They emphasise the importance of supporting provenance capture throughout all stages.

To capture retrospective provenance of script-based workflows, commonly utilised and prominent efforts include StarFlow [121], noWorkflow [122] and YesWorkflow [123]. StarFlow is a Python-specific data analysis environment which operates by inspecting file dependencies and function-level control flows using static analysis of the code and dynamic analysis during execution. noWorkflow is proposed to capture "deployment", "definition" and "execution" provenance from the execution of Python scripts without requiring the scientists to modify their existing scripts. Thus, noWorkflow focuses on capturing retrospective provenance for each execution of a given Python script.

YesWorkflow also does not require users to change their scripts but rather annotate them with specialised comments. These comments are then analysed by YesWorkflow to identify the computational needs of these scripts and pro-

duce their "workflow-like" graphical representation. YesWorkflow is more focused on the prospective view of provenance and is language independent such that any scripting language can be coupled with the YesWorkflow annotations to generate explicit graphical representation of the workflow from these scripts. In some studies [124, 125], the authors utilised the combination of noWorkflow and YesWorkflow to benefit from the unique characteristics of both solutions to capture both retrospective and prospective provenance.

2.2.5.2 Workflow Management Systems with Built-in Provenance Capture

Workflow Management Systems (WMS) defined as software environments are used extensively by researchers to design and execute complex workflow-centric analyses [126]. Typically, any WMS is able to support the design of workflows by utilising existing resources such as data and configured software or by seamlessly incorporating new datasets and tools. Such systems facilitate the execution of workflows through user-friendly graphical representations and intuitive designs [127]. Some example WMS used extensively for scientific workflow design and execution include Taverna [5], Kepler [7], VisTrails [6], Wings [113] and Galaxy [8].

Analysing WMS evolution in the past decade, Atkinson *et al.* [127] list a range of characteristics that support the re-usability of the methods and results, automation of processes and sharing of workflows across the scientific community. Provenance capture and management has been identified as a key characteristic for WMS to support for many applications domains (discussed in Section 2.2.3) and especially with respect to scientific workflows. Despite the common goals and characteristics of all WMS, a *"one-size-fits-all"* approach for provenance capture mechanisms across such systems does not exist. For example Galaxy follows

its own approach for provenance capture by saving and representing provenance in the form of either "histories" or "pages", both of which can be shared using a public link or by publishing through the Galaxy platform for broader community use. On the other hand, Taverna developers have been actively participating in the provenance standard building activities described in Section 2.2.4. Hence, Taverna has an extensive provenance capture and management system with capabilities to export this information to the W3C Provenance Ontology [128]. Detailed analysis with respect to provenance capture and management of the existing WMS used for scientific workflow design and execution are included in Section 2.4.

2.2.5.3 Standalone Tools for capture

The WMS-based capture of provenance requires appropriate provenance-aware design choices to be made when WMS is developed, to avoid adding provenance capture support retrospectively. The script-based provenance capture is usually considered as an option by researchers if excessive re-factoring of the pre-built pipelines is not required or if the provenance-centric script code can be incorporated into the pipelines during their development. Another common approach to provenance capture is incorporating independently designed standalone tools that operate on existing workflows written in high-level scripting or workflow definition languages. Such tools require minimal or no change to the WMSs or scripts to enable provenance information collection of already built WMS/scripts.

Standalone provenance capture tools can operate and obtain provenance information in different ways, e.g. exploiting operating-system level information to obtain a high-level view of provenance [129, 130, 131]; by modifying the software to enable generation of the required fine-grained provenance information [132,

133], or by implicitly capturing provenance information by identifying data flows [134]. Such efforts can also be language-specific. Examples of such approaches include Sumatra [135], ProvenanceCurious [136] and RDataTracker [137]. SHARP [138] is a more recent effort to harmonise provenance capture across two platforms (Galaxy and Taverna), where the authors have developed an open-source tool to extract provenance information from a given Galaxy workflow and represent the results in PROV RDF triples which can be used in combination with the provenance graphs generated by Taverna to run cross-workflow queries.

2.2.6 Provenance Data Analytics

Provenance data analytics deals with querying, mining and extracting knowledge from potentially voluminous amounts of provenance traces generated following the execution of large-scale workflow-based experiments. This section covers mainstream approaches to tackling provenance-based data analytics.

One common approach is utilising provenance graphs to determine data quality or identify domain-specific information. Keshavarz *et al.* [139] present an analytic approach focused on traversing provenance graphs and utilising the available annotations to assess data quality and reliability for run-time decision making regarding task continuation or termination. Huynh *et al.* [140] leveraged the concept of descriptive analysis of network graph characteristics and proposed twenty-two provenance network metrics that could be used to answer provenance-related data queries. The authors applied supervised learning methods to provenance graphs to predict domain-specific information.

Data mining techniques have also been used as methods to simplify and better represent provenance graphs. Chen *et al.* [141] reduce the feature space of

provenance data sets by dividing the OPM-graph into a sorted partition to generate a reduced-sized temporal representation of the provenance data thereby providing a high level and more understandable view. Another technique of summarising provenance is discussed by Moreau [142] which focused on generating a summary of provenance information by grouping similar provenance types to increase the understandability of the provenance data. He defines and satisfies the key requirements that any summarization technique should consider, i.e. the summary should capture the *"Essence of Provenance"* and it should be evident whether there is *"Conformance"* with the provenance graphs. A summary should also be able to detect *"Outliers"*.

A recent survey by Oliveira *et al.* [143] focused on the analytics aspect of workflow-based provenance. The authors describe current computational methods commonly utilised for analysing provenance data for information extraction; for improved understandability, and for subsequent usability of this data. These methods include **data mining**, **summarization** of provenance graphs, **customisation** of provenance views, syntactic and semantic **inference** and **similarity comparison** to identify evolution of data and methods used in workflows. The characterisation by Oliveira *et al.* is used as reference in the taxonomy devised in Section 2.2.8.

2.2.7 Resources Supporting Provenance Applications

The applications of provenance discussed in Section 2.2.3 will not be supported ubiquitously if the only resource shared is a provenance document and a trace of a given workflow enactment. Such traces contain references to the resources used, e.g. data artefacts, computational tools, workflow specifications and configuration settings. Hence, sharing and publishing these related resources is of

paramount importance to enable best use/re-use of the provenance information.

Cohen at al. [144] and Belhajjame *et al.* [29] also emphasise that aggregating, packaging and publishing the computational methods and data utilised in a given workflow enactment are essential. This includes the run-time environment, data artefacts, workflow specifications and provenance trace of the workflow enactment. Often, emphasis is placed on provision of these resources without explicitly mentioning the effect of their absence on the application. Hurley *et al.* [145] also emphasised that sharing "recipes" of workflow ingredients (satisfying the requirements of prospective provenance) and "snapshots", e.g. VMs and cloud instances, with minimal implementation of the software required to reproduce the published analyses or verify the results is key. The following subsections discuss the conventional approaches utilised to support resource sharing.

2.2.7.1 Workflow Software Environment Capture

Freezing and packaging the run-time environment to encompass all software components and their dependencies used in an analysis is a recommended and widely adopted practice [144] especially for cloud computing resources where images and snapshots of cloud instances can be created and shared with researchers [146]. The software packaging and availability alongside published analysis streamlines the process of evaluation and re-use of the shared resources by reducing the efforts to manually manage complicated configuration settings. This can help to minimise the effects of complex software dependencies. There are many approaches to packaging software environments. We consider categories based on the nature of their functionality and resulting costs of overheads on the part of the user.

2.2.7.1.1 Virtual Machine & Cloud Instance Snapshots

Hypervisor-based virtualisation technology has been around for a long time. It caters for the needs that often arise in distributed computing including configuration independence, resource distribution and software interoperability [147]. Generating snapshots to encapsulate the current state of a computing environment is referred to as *"Whole system snapshot exchange"* [148] and *"System-wide Packaging"* [144]. Kernel-based Virtual Machine (KVM), VirtualBox, VMWare Server and Vagrant are a few of the existing virtualisation solutions for packaging and sharing workflow software environments that allow to create the exact replica of the required environment.

Cloud computing has emerged as a predominant solution to manage and support shareable software environments by providing *"Infrastructure as a Service (IaaS)* [149, 150]. These snapshots can not only capture the computational environment but also the datasets and other resources that can subsequently be distributed. Pre-configured snapshots can be used as a starting point to launch cloud instances with enough compute and storage resources to validate the shared analyses, reproduce results and/or re-use shared methods across studies. Leading platforms managing infrastructure and providing cloud computing services and configuration on demand include DigitalOcean [151], Amazon Elastic Compute Cloud (EC2) [152], Google Cloud Platform [153] and Microsoft Azure [154]. The instances launched on these platforms can be saved as snapshots and publicly published with a given analysis. This allows to create instances representing the computing state of the snapshot, at a given future date.

"System-wide packaging" for data-driven analyses, although simplest for workflow developers has its own caveats. One notable issue is the size of the snapshot as it captures everything in the instance at a given time, hence the size can range from a few Gigabytes to many Terabytes. Furthermore, snapshots lack contex-

tual and background information about the underlying environment and must be accompanied with the provenance information, e.g. to provide the scientific background and explanation of the shared resources [28].

2.2.7.1.2 Container-based Solutions

To address the issues related to size of instance snapshots, platform differences and conflicting dependencies, platform-independent light-weight LinuX containers (LXC) are emerging as a new choice for the "unit of deployment" [155]. These containers define a computing environment capturing the dependencies of software by virtualisation at the Operating System (OS) level and utilising existing resources of the host environment. These containers should encapsulate an immutable version of an application coupled with the minimal OS components essential for execution of that application. Gruening *et al.* [156] identify recommendations specifically for the bioinformatics domain to make the software involved in an analysis discoverable, re-usable and transparent. The authors recommended use of dependency managers along with light-weight containers to achieve preservation and availability of software applications that are utilised in a given analysis.

Beaulieu & Greene [157] suggested the use of container-based technologies such as Docker along with continuous integration techniques to facilitate preservation of methods utilised in data-intensive analyses where a full service continuous integration service is configured to re-run the analysis in case of changes in source code or data. In addition to Docker, other current leading container solutions including LXD, Singularity [158], rkt[9] and BioContainers. Each of these have unique characteristics making them an appropriate choice in different use-cases. Docker also allows to publish and share images with the scientific commu-

[9] https://coreos.com/rkt

nity via an image store, Docker Hub registry[10]. Singularity is considered suitable for High Performance Computing (HPC) with SLURM and UGE like workload managers. It also enables portability of Docker images to Singularity to allow re-use of existing Docker containers in other environments.

Notable open-source community-driven efforts include domain-specific registries such as BioContainers[11] and Dockstore[12]. These allow for the preservation and sharing of executable environments with the help of Docker and rkt containers.

2.2.7.1.3 Package & Configuration Management

Another common practice adopted by the scientific community to ensure seamless availability of computational methods is through configuration and package managers. The dependencies of a given application include libraries, applications and metadata are often aggregated as a single "software package" [156], which can be deployed by configuration and package managers as and when required.

These package and configuration managers not only handle one-time installations but also provide facilities to upgrade pre-installed packages. Such systems can operate at the Operating-System level and handle centralised installations such as apt, yum, make and pkg. Other scripting language-specific solutions to package management include pip and PyPI for Python, Packrat[13] and CRAN[14] for R and CPAN[15] for Perl. Operating-System solutions such as MacPorts and Chocolatey for Windows are also commonly used package managers nowadays. When choosing such a system, it is recommended that a platform-independent

[10] https://hub.docker.com/
[11] https://biocontainers.pro/
[12] https://dockstore.org/
[13] http://rstudio.github.io/packrat/
[14] https://cran.r-project.org/
[15] https://www.cpan.org/

solution that enables installation of multiple versions of an application is chosen. Those solutions that do not require administrative privileges should be given preference [156]. Such features promote interoperability and simplified usability by giving the user control over the computational environment without requiring administrative rights to make changes to the system. Examples of such third-party standalone systems include HomeBrew/HomeBrew Cask[16], Conda[17], Nix[18] and Zero Install[19].

Community-driven bioinformatics-specific efforts have recently adopted Conda as the preferred mode of packaging, due to its language-agnostic and platform-independent nature. This makes it suitable for both HPC and cloud infrastructure deployments [159]. Bioconda [160] is based on Conda and comprises of bioinformatics installation recipes, a build system to convert these recipes to packages and a repository offering more than 3000 bioinformatics packages. Each Bioconda package can be installed via Conda, Docker, rkt and Singularity making it highly portable.

2.2.7.2 Data/Method Preservation, Aggregation & Sharing

The workflow resources e.g. the workflow specifications, command line tool specifications, datasets used in an analysis and underlying software are required to be aggregated and packaged for each workflow-based analysis, preserved and published alongside conventional publications to address various applications of provenance. Expecting individuals to address this issue would (and does) lead to disconnected, non-standardised efforts resulting in numerous platform and

[16]https://brew.sh/
[17]https://conda.io/
[18]https://nixos.org/nix/
[19]https://0install.net/

application-specific solutions. In response, subsets of the scientific community increasingly drive efforts to define unified formats of aggregation of resources and open-access sharing repositories for publishing such artefacts. In this section, we review some of the major efforts in this area.

2.2.7.2.1 Structured Representation and Aggregation

There are several initiatives focusing on preservation of the digital artefacts utilised in a workflow enactment [161]. This encompasses the hypothesis tested in the given analysis, the findings, the meta-data that does not qualify as provenance, retrospective provenance of the given workflow enactment, prospective provenance and most importantly machine-readable annotations explaining all of the aggregated resources and their relationship. These annotations provide contextual information about the exact role an artefact plays in the research lifecycle of a given workflow enactment. Critically, this is more than a zipped folder of all the artefacts related to an analysis, but rather it should comprise a well-formatted and structured description of the experimental process.

Pioneering resource-centric initiatives using semantic technologies for systematic data and method aggregation include the Open Archives Initiative Object Re-Use & Exchange (OAI-ORE) [26], the Investigate Study Analysis (ISA) framework [162] and the computational biology-specific COMBINE Open Modeling EXchange format (OMEX) Archive [163]. These standards and frameworks are utilised by the scientific linked data community to tackle workflow-oriented studies covering domain and platform-independent Research Objects (RO) [29], self-contained provenance aware packages such as ReproZip [28], BioCompute Objects [164] specific to bioinformatics and provenance-rich DataOne packages [27]. A common goal of these self-describing bundles is to promote digital preservation, portability and interoperability of compute-intensive scientific experiments.

This thesis utilises Research Objects specific to workflows following the BDBag approach, which itself is based on *BagIt* [165], for method and data aggregation of a given workflow enactment. Details of the principles justifying the adoption of these standards are presented in *Chapter 5*. Several platform-dependent studies have also been produced as extensions to existing standards by implementing RO concepts and improving aggregation of resources.

Belhajjame *et al.* [29] proposed the application of ROs to develop workflow-centric ROs containing data and metadata to improve the understandability of utilised methods (in this case workflow specifications). They explored five essential requirements to workflow preservation and identified data and metadata that should be stored to satisfy the said requirements. Specifically, they proposed extensions to existing ontologies such as Object Reuse and Exchange (ORE), the Annotation Ontology (AO) and the PROV-O ontology and also developed four new ontologies to represent workflow specific information. However, as they identify, the scope of the proposed model at that time was not focused on interoperability of heterogeneous workflows but on a given Taverna Workflow Management System using myExperiment, which made the approach platform-dependent.

Gomez *et al.* [166] proposed extensions to the RO model augmenting workflow-centric ROs with information to cater for the specific needs of the Earth Science community. They demonstrated that ROs could support extensions to generate aggregated resources leveraging domain specific knowledge. Hettne *et al.* [167] used three workflow case studies in the genomics domain to demonstrate how Research Objects could capture methods and data supporting subsequent querying and extraction of information about the scientific investigation under observation. The solution was again tightly coupled with the Taverna Workflow Management System and hence if shared, would not be interoperable or reproducible outside of the Taverna environment. Other notable efforts to use Research Ob-

jects for workflow-centric studies include [5] in systems biology, [168] in clinical settings and [169] in precision medicine.

2.2.7.2.2 Preservation and Sharing via Repositories

The sharing and publishing of resources linked to scholarly publications is now supported by various online repositories. Various journals such Nature, Biostatistics, PeerJ and Science mandate that data and computational methods are shared in publicly accessible repositories such as GitHub, Dataverse[20] and re3data, while others such as GigaScience take this a step further by providing their own repository, GigaDB [170] for sharing data and methods utilised in work under consideration for publication. WMS-specific initiatives such as Galaxy with histories and toolsheds along with generic efforts such as myExperiment [171] and Dataverse [172] offer workflow-centric options for open access method sharing. Some other commonly used open source online repositories for collaborative research include Zenodo[21], GitHub[22] and Figshare[23].

Such resources facilitate collaborative research in addition to sharing of source code underpinning a given analysis. There is no standard format defined or declared that must be followed when someone shares artefacts associated with an analysis. As a result, the quality of shared resources can range from a highly annotated, rigorously documented and complete set of artefacts, to raw data with undocumented code and incomplete information about the infrastructure as a whole. Individual organisations or groups might provide a set of "recommended practices", e.g. in README files that attempt to maintain the quality of shared resources. An exemplar initiative *Code as a Research Object*[24] is a joint project be-

[20] https://dataverse.org/
[21] https://zenodo.org/
[22] https://github.com/
[23] https://figshare.com/
[24] http://mozillascience.github.io/code-research-object/

tween Figshare, GitHub and Mozilla Science Lab that aims to archive any GitHub code repository to Figshare and produce a Digital Object Identifier (DOI) to improve the discovery of resources.

2.2.8 Putting the Pieces Together

We have defined and introduced the key concepts of provenance with a focus on scientific workflows, its related principles and supporting resources which are revisited frequently in later chapters. In this section, we present a taxonomy encompassing these elements that is explored and categorised in the future sections. This taxonomy is used to evaluate state of the art Workflow Management Systems and workflow definition approaches as categorised in Section 2.3.2.

A consolidated view of the provenance concepts in the form of a taxonomy was presented by Simmhan *et al.* [45]. This overlaps slightly with respect to the applications of provenance described in Section 2.2.3. Another detailed survey of provenance for workflow management systems focusing on provenance capture mechanism, classes, storage and query support was presented by Freire *et al.* [173]. The provenance capture mechanisms described in Section 2.2.5 partially overlap with the metrics used in this survey in one key aspect - namely "Workflow Management System built-in Capture". The representation of provenance analytics methods in our taxonomy are adopted from another notable survey by Oliveira *et al.* [143] presented in Section 2.2.6.

The taxonomy organisation in this thesis as shown in Figure 2.7 extends already defined categorised provenance concepts, but with focus on scientific workflow execution provenance and its supporting resources. The argument to include these resources in the taxonomy is based on their utility, i.e. without them

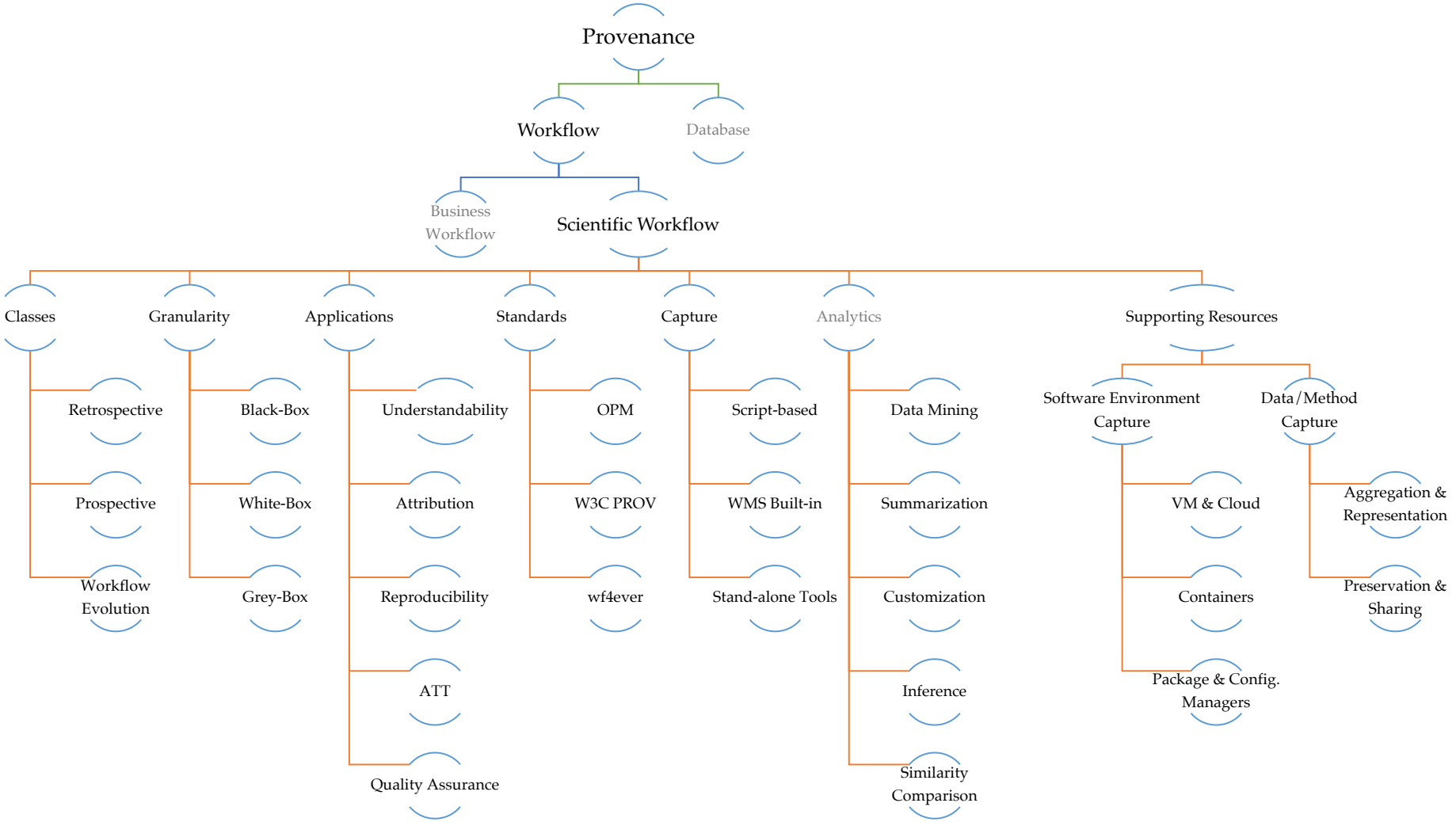

Figure 2.7: Scientific Workflow Provenance Taxonomy (Grey nodes are out of scope of this thesis)

the captured provenance would be unable to serve the primary purpose it is captured for.

2.3 Workflow Definition & Implementation Approaches

Scientific workflow design and management has become an essential part of many computationally driven data-intensive analyses enabling Automation, Scaling, Adaptation and Provenance support (ASAP) [3]. Varying requirements to be met by workflows and their increased use has driven rapid growth in the number of computational data analysis workflow management systems, with hundreds of heterogeneous approaches now existing for workflow specification and execution [174, 175].

In this section, the evolution of systems for the definition and execution of scientific workflows is presented. Following this, the various heterogeneous workflow definition approaches used in the bioinformatics domain are categorised into three broad categories.

2.3.1 Evolution of Workflow Management Systems

Workflow Management Systems (WMS), as mentioned in Section 2.2.5.2, enable researchers to perform data-intensive analyses by systematically supporting the creation and execution of workflows by chaining together computational tools and data sources. In theory, all WMSs are expected to capture the exact methodology used in a given workflow-centric analysis, ideally capturing and re-using the associated provenance information needed to support reproducibility and attribution. In practice, this capture and re-use varies considerably and if it exists,

then it is specific to each WMS.

Starting from largely isolated solutions a decade ago, the trend of research and workflow-oriented support has shifted to encourage abstraction in workflow definition. Recent efforts have focused on the creation of interoperable and portable methods that can be executed on multiple platforms and utilising varying computing resources. This abstraction enhances the understandability of workflows and allows for different expertise levels of researchers to avoid the steep learning curve and computing knowledge that was historically required.

With the growing computing capacity and ever-evolving computing platform architectures now available, WMSs have evolved to enhance performance by leveraging diverse computing resources. Atkinson *et al.* [127] credits parallel computing, scheduling and planning, and data management as key factors that have improved WMSs. Various workflow-oriented studies [176, 177] have focused on improving the performance of scientific workflows by applying techniques to better schedule workflows to optimise use of available computing resources.

The *"Open Science"* movement has the goal to enhance research performance by establishing the foundations of research based on openness and sharing. This has also resulted in community driven WMSs (described in Section 2.4.1) that are not grant-dependent for sustainability, as is the case for much research software. The sharing of resources such as workflows and their associated data has become a new norm for improving the trust of published studies by making them transparent and reproducible. To handle this, WMSs now facilitate the sharing of workflow resources by utilising many of the tools described in Section 2.2.7.1 to distribute information leading to an increasingly decentralised view of computation and data.

2.3.2 Classification of Workflow Approaches

Out of the many big data domains, genomics is considered *"the most demanding"* with respect to all stages of the data lifecycle - from acquisition, storage, distribution and analysis [178]. Genomic data is growing at an unprecedented rate due to improved sequencing technologies [179] and reduced cost. It is currently challenging to analyse such data and this challenge is expected to grow for the foreseeable future. *in silico* experiments designed for analysis of such data require execution of various local applications and remote services connected with each other in the order of their execution. In this context, scientific workflows have overtaken many traditional research methods using ad-hoc scripts, which has been the typical modus operandi over the last few decades in bioinformatics domain [127, 180].

In this section, we classify key approaches for workflow definition and implementation[25] into three broad categories as shown in Figure 2.8. These categories have been identified based on the most common practices prevailing in bioinformatics workflows design and management. This characterisation has been published as part of an empirical case study detailed in *Chapter 3*. Although this classification holds for any domain utilising workflow-centric computational tools for data analyses, we focus especially on bioinformatics-specific platforms.

2.3.2.1 Domain-specific Pre-built Pipelines

Several automated bioinformatics-specific pipelines such as Cpipe [181], bcbio-nextgen [182] and others [183, 184] have been developed using command-line

[25]This classification has been published in the following article:
Kanwal, Sehrish and **Khan, Farah Zaib** and Lonie, Andrew and Sinnott, Richard O **Investigating reproducibility and tracking provenance–A genomic workflow case study.** In: BMC bioinformatics 18.1 (2017), p. 337.

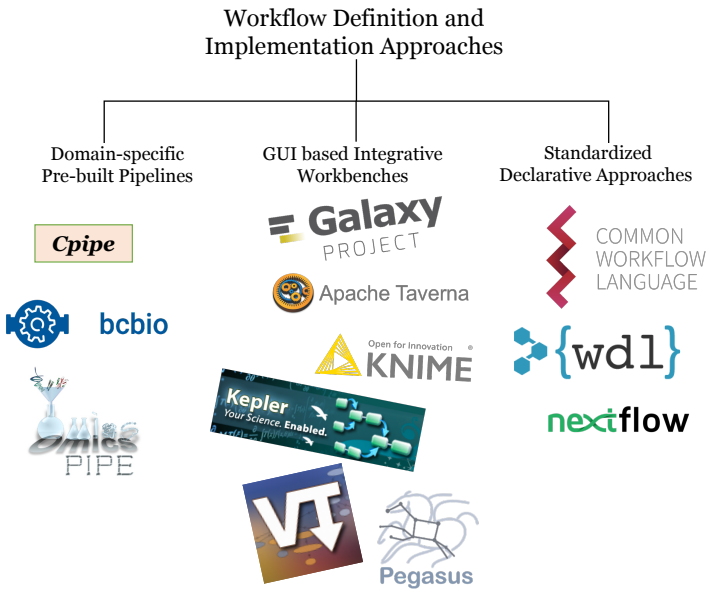

Figure 2.8: Classification of workflow definition approaches used in the bioinformatics domain with exemplar solutions for each category.

tools to support genomic data analysis. These pipelines are typically driven and supported by individual laboratories, which have developed customised pipelines for processing data. This approach has resulted in considerable variability in the methods used for data interpretation and processing. The advantages of these pipelines include editing pipelines on remote servers without requiring access to graphical user interfaces (GUIs). This means that they are easily administered through source code management tools [185]. However, command-line based pipeline frameworks such as bpipe [186], Snakemake [185] and Ruffus [187] are often not flexible enough to support integration of new user-defined steps and analysis tools.

Furthermore, working with such systems requires expertise with command-line programming and typically deep computational knowledge as these systems extensively use individual scripts to tie together different components of the pipelines. These scripts control variables, dependencies and conditional logic

to provide efficient processing of the data, however they are often difficult to reproduce. Such systems often assume the provision of the same physical or virtualised infrastructure used to run the initial analysis, including scripts, test data, tools, reference data and databases. The implementation overheads of such pipelines include configuration and installation of software packages, parameter settings and their potential alteration, debugging and lack of input/output interfaces. As such, considerable effort and time is required to create, understand and reproduce such pipelines.

2.3.2.2 Graphical User Interface (GUI)-based Integrative Workbenches

To tackle some of the challenges of pipelines created using command-line solutions, workbenches such as Galaxy [8], Taverna [5], Kepler [188], VisTrails [6], Wings [113], Pegasus[189] and Knime [190] have been developed. These support easy and customised workflow definitions using a GUI-based front end. Few of the referenced workbenches allow researchers to specify the goals, requirements and constraints for workflows using for example semantic reasoning [191]. Semantic workflow management systems support setting up analyses by providing parameter preferences, alternate software tools and relevant datasets whilst leveraging analytic constraints articulated by the user [192]. The semantic descriptions typically expect complex validation rules for input and output data objects to support more advanced reasoning. Hitherto such systems havent been widely adopted because of the complications involved in modelling systems, the rapid evolution of semantic web services and the fact that the majority of existing approaches adopt a non-semantic approach [193].

GUI-based workbenches are typically highly featured and pre-configured with modular tools supporting interactive design suitable for a wide range of audi-

ences with varying degrees of expertise. Such environments often include auxiliary datasets and configuration settings to aid users in designing automated and robust pipelines that provide managed access to a library of systems with abstractions of the interaction layer and a workflow layer that captures tool versions and parameter information. Such GUI workbenches can be easily used with already existing tools but adding a new tool or executable wrapper can require an in-depth familiarity of acceptable input file types, parameter settings, exception handling and resource management.

However, such systems do not require any local installations of the analysis tools nor customisation of the analysis environment, hence they have lower infrastructure maintenance costs. On the other hand, the availability of external services and customised tool repositories poses a risk to reproducibility as it is typically impossible to reproduce a workflow created using a service or external data resource which has been changed or is no longer available. Similarly workflows implemented on one system may not be reproducible when imported into another system due to incompatibility issues between local customised environments.

2.3.2.3 Standardised & Declarative Approach to Workflow Definition

The heterogeneity in the field of *in silico* genomic analysis has motivated researchers to work towards standardised workflow description languages. Examples of these include the Common Workflow Language (CWL) [31] and the Workflow Definition Language (WDL) [194]. A variety of software platforms, such as individual workstations to high performance computing platforms (cloud, grids or clusters) can be used to enact workflows. Workflows should ideally provide a formal specification of the required environment, covering all aspects of the

workflow implementation including tool versions, input data, customizable parameter settings and the workflow run-time environment. Ideally such approaches provide software specifications that help researchers define and implement portable, easy to use and reproducible workflows. The specifications should describe data and offer execution models allowing users to have full control when creating and running workflows by explicitly declaring the relevant environment, resources and other customizable settings in addition to the workflow specification.

2.4 Evaluation of Exemplar Approaches

In this section, we introduce and evaluate exemplar systems from each category of workflow definition approaches with emphasis on their features for provenance capture and use. There are a multitude of existing systems [174] and this section does not aim to achieve completeness in evaluation of all existing systems. Rather, the systems are mainly selected based on their use in bioinformatics research, their principled approach to provenance capture or the tools they leverage to capture the software environment required to enact the workflow.

2.4.1 Exemplar Bioinformatics-specific Pre-built Pipelines

There are many other pre-built pipelines used in the clinical domain to support high throughout bioinformatics data analysis. Notable projects include NGSane [195], Targeted REsequencing Virtual Appliance (TREVA) [196], NGS-pipe [197] and DNAp [198].

2.4.1.1 Cpipe

Cpipe provides a consensus pipeline designed for clinical diagnostics. For provenance recording and provision, Cpipe provides a minimal view in the form of a human-readable PDF document[26] as a "Provenance Report". This contains three tables: 'Pipeline', 'Tools' and 'File' separating the details of the pipeline version, researcher responsibilities to enact the pipeline and the underlying tool versions and file names used in the analysis. This view is quite coarse-grained and lacks a standard representation of the details of the derivation history, i.e. it has minimal support for retrospective provenance.

Cpipe uses a programmatic approach which effectively includes everything necessary for reproducing a given genomic analysis. It expects the same/equivalent physical infrastructure and software environment used to run the initial analysis, including scripts, test data, tools, reference data and databases. For software environment capture, virtualisation technology in the form of a cloud instance snapshot is typically utilised to allow users to launch a new instance with Cpipe [199]. Further details of the functionality and inner workings of this pipeline are presented in *Chapter 3*.

2.4.1.2 Bcbio-nextgen

Bcbio-nextgen is a community-developed and community-driven effort comprising a diverse collection of pre-built pipelines utilising best practice approaches to realise biological tasks such as Germline Variant calling, Somatic Variant calling, RNA-seq and ChIP-seq data analysis. A pre-defined test/example dataset and known outputs of each pipeline is available to support validation of newer

[26] https://github.com/FarahZKhan/GATK-CaseStudy/blob/master/Cpipe/Cpipe-prov.pdf

methods such as a new version or a different algorithm [200]. The pipelines are scalable across heterogeneous computing platforms and designed to be compatible with research-oriented analyses including population-wide and individual genomes.

Bcbio-nextgen mostly relies on and expects to utilise the capabilities of the execution platform (such as Arvados [201]) for provenance capture. That is, there is no standard representation for provenance capture. Rather, it varies depending on the execution platform resulting in considerable heterogeneity of the provenance of results despite use of the same pipelines. Hence, the granularity of complete provenance depends on the choice of platform and the underlying provenance capabilities it offers.

Regarding the software environment, it is recommended to create a virtual Python environment to run an automated script provided for installing the necessary resources. A Docker file is also available for initial configuration of the packages required for the Bcbio-nextgen framework, thereby minimising manual time-consuming configuration. The resources linked to any pipeline including source code, Docker images and test data are freely available and shared in different open-access repositories.

2.4.1.3 Omics-Pipe

Omics-Pipe is another open-source pre-built Python-based pipeline. It utilises Ruffus, Sumatra [135] and C libraries for running processes, offering version control and utilising distributed computing resources. It is modular and extensible in nature allowing researchers to write Python modules that can be incorporated in the pipelines.

Provenance tracking for the Omics-Pipe execution relies mostly on informally capturing metadata as supported by Sumatra. This metadata includes details of the hardware platform, the source code of the pipeline executed, the versions of input datasets, the configuration setting of tools and the output data that is linked to each execution. Hence, without characterising and formally representing the captured metadata, each pipeline execution produces elements of retrospective provenance. It is common practice to change the configuration settings of any/all tools for result comparison, hence Sumatra associates each run with the version of the pipeline and the associated configuration settings used.

An Amazon EC2 instance image with pre-installed dependencies is made available to be run on specific clusters. A Docker container is also mentioned but no information is provided about its availability or access. The source code is freely available on Bitbucket for local installations in which case the user is expected to manually install any/all third party tools and databases and deal with the dependency issues that may arise.

2.4.2 Exemplar GUI-based Workbenches

There are many GUI-based WMSs extensively used in bioinformatics research. Some of these are community-driven in terms of methods used and the workflow development and sharing. We focus here on their support for retrospective provenance.

2.4.2.1 Galaxy

Galaxy is an open source web-based platform. It is extensively used for data-intensive biomedical research with more than 5,700 citations and 500 developers maintaining and improving the platform [8]. It is one of the most accepted and adopted workbenches used by researchers with a strong community contributing towards building and improving the quality of methods and ultimately, the research it supports.

As described in Section 2.2.5.2, Galaxy has it's own approach for provenance support. This is not based on use of convention or standards for representation of such information. For each workflow execution, Galaxy uses a view of "History" that automatically tracks changes made to the data files and tools used to derive a given data product. It also provides a manual annotation facility to add user-defined "tags" to enhance understandability of file names or by adding domain-specific information. Once researchers are satisfied with a workflow execution to be shared, a Galaxy "page" can be developed incorporating data, tools and context-specific details such as the hypothesis tested, attribution and justifications for use of specific tools or configuration settings. Hence, some retrospective provenance elements are available but these are largely scattered across the platform without a unified view or standard representation.

For preservation of the software environment, the Galaxy community has recently adopted package managers such as Conda and Bioconda to resolve software dependency issues. Bioconda is a community-driven project with >2,700 bioinformatics packages available as Docker containers ready to be installed offering specific software versions. There also exists a centralised Galaxy ToolShed repository [202] that is used to disseminate software and software dependencies across different public Galaxy instances.

2.4.2.2 Taverna

Taverna is a one of the pioneer open source WMS supporting the coupling of distributed web-services into workflows such that any researcher can utilise the existing services without requiring them to manually install tools.

Taverna is a community-driven effort and included features for provenance capture and re-use. It enables detailed provenance collection of workflow enactments using the Taverna Provenance Suite and allows exporting the provenance information to OPM and W3C PROV models. It offers Research Object support and used of standards such as *"wfdesc"* for prospective provenance and *"roevo"* for workflow evolution representation.

As a platform Taverna introduced the Research Object notion [5] for workflow and method aggregation. It leveraged myExperiment repository [203] for preservation of the workflow along with example input data and sharing with the community. However, shared workflows depending on third-party resources such as external web-services introduces challenges due to the potential unavailability of these resources at a given time, resulting in *workflow decay*. In this case, users have to provide alternatives for the required web-services, e.g. from a service registry such as BioCatalogue [204].

2.4.2.3 Kepler

Kepler is an open source Java-based cross-platform WMS. It is one of the most highly cited workflow-oriented systems spanning various scientific domains[27] including bioinformatics [205]. It has a dedicated module for the large-scale analysis of biological data in distributed environments [206]. It can utilise Hadoop,

[27] https://kepler-project.org/users/projects-using-kepler

Spark or the Stratosphere platforms to handle larger scale genomic data. It also supports aspects of provenance through *"smart re-runs"*.

As with Taverna, the Kepler authors were highly active in provenance challenges and made significant contributions towards improving the understanding of workflow-specific provenance. Kepler includes a dedicated optional layer for provenance capture and storing the context of the research. This includes support for input data and metadata, the outputs of each enactment, the workflow definitions utilised and the context of the workflow enactment including the attribution details given in PROV JSON format [207]. In addition, cross-enactment parameter space comparison is also included in provenance traces. All information is stored in a database which can be queried using an API.

Like Taverna, dependence on third party resources such as remote data and tool access during a workflow enactment poses risk because of the ever-changing resources used over the web. However, there is no mention of capturing and exporting the software environment needed for a workflow enactment when an analysis is shared and expected to leverage provenance.

2.4.2.4 Pegasus

Pegasus is another widely used WMS doing big data computations utilising Map-Reduce, Storm and/or Spark for big data computations. It also enables collaborative science with its cross-platform capabilities including use of local systems, clusters and cloud resources [208].

Retrospective provenance in Pegasus is referred to as *"job runtime"* provenance. This is captured by a separate executor on distributed nodes *pegasus-kickstart* [209] (if distributed resources are used). This is sent back along with the

results and subsequently stored in a database. This information includes the parameter space, start/end time of a process, logging information and the machine information where the process was executed. However, this information gives a Pegasus-specific view and lacks various capabilities, e.g. the ability to export to a format conformant to the standardised provenance models. In addition, Pegasus abstracts the workflow descriptions in XML format to achieve portability across different platforms.

Although support for cluster and cloud computing resources is available in Pegasus, such practice is uncommon in the workflow implementations. The WMS itself is available as a light-weight Docker container enabling easy installation for researchers. Recently, there have been a variety of efforts[28] to incorporate container technologies to support software configuration. In addition, the Pegasus Jupyter Python API supports the use of Pegasus via Jupyter notebooks [210] to capture the reproducibility requirements in a single place. However, this approach is still in its embryonic stage and not yet widely adopted.

2.4.2.5 VisTrails

VisTrails is an open source multi-platform Python-based WMS with front end GUI to execute workflows interactively. It also has command-line support for running jobs in batches using the VisTrails server.

VisTrails comes with an innate comprehensive provenance infrastructure as part of its Provenance Software Development Kit (ProvSDK). The generated information can be exported either in XML format for sharing or kept in a relational database for querying. The VisTrails design supports exploratory computational research by enabling users to view past executions of a workflow and

[28]https://pegasus.isi.edu/2018/02/05/cyverse-container-camp/

iteratively improve the configuration of these workflows to attain the required results. This approach treats workflow specifications as data and captures the elements of workflow evolution. This includes workflow refinements over time such as different configuration settings, adding/subtracting modules or updating versions of existing tools. Prospective provenance is represented as Python objects which are available for export, however retrospective provenance is only available through the relational database. It is noted that the native schema of VisTrails provenance only partially overlaps with the W3C PROV notations [211].

VisTrails originally introduced the notion of incorporating workflows in a publication by providing a LaTex package which enables users to add their results through charts or graphs in a document. These results when clicked, retrieve the workflow used to produce the results. Workflows to be included in the documents are recommended to be shared on CrowdLab [212], an open-access repository for sharing workflows, together with results of the enactment. To reenact the workflow, the user still needs to handle manual installations of related resources, e.g. the tools required in the workflow. Recent efforts[29] have looked at use of Docker for the configuration of VisTrails together with Jupyter notebooks, however such approaches are not yet in mainstream use.

2.4.2.6 BioWorkbench

BioWorkbench [213] is a recent cross-platform WMS specific to bionformatics experiments. It uses Swift [214] at its core for workflow construction and for provenance collection.

BioWorkbench has a layered architecture separating Specification & Execution, Data and Analytics operations. The data layer supports extensive prove-

[29] https://hub.docker.com/r/vidanyu/vistrails/

nance collection from logs generated by each execution of the workflow. Currently the generated provenance is stored in a relational (SQLite) database with the capability to export it to OPM format. The Analytics layer processes the generated provenance data by applying machine learning algorithms using Weka [215] for knowledge extraction to optimise workflow enactments by predicting computational resource requirements.

To meet framework availability and configuration requirements, the use of Docker[30] is highlighted. However, it is not evident if the individual tools utilised in a workflow can also use Docker or if tools used in one workflow are freely available in a centralised repository that can be be utilised by other researchers.

2.4.3 Exemplar Standardised & Declarative Approaches

There have been several recent and rapidly endorsed declarative workflow standards in bioinformatics domain. The theme of these approaches is to design workflows as abstractly as possible to improve their portability without being tied to a single platform. The SHaring Interoperable Workflows for large-scale scientific simulations on Available DCIs (SHIWA) project [216] was designed with similar goals of interoperability and portability of workflows, however it lacked a strong research (community) base and was not sustained. We consider mainstream approaches here.

[30]https://hub.docker.com/r/malumondelli/bioworkbench/

2.4.3.1 Common Workflow Language (CWL)

CWL is an open-source community developed and maintained standard for designing declarative descriptions of computational data analysis workflows with constructs to support sub-workflows and job distribution. With its rapid acceptance by many participating organisations together with support from existing cloud-enabled WMSs such as Galaxy, Toil [217], Taverna and Arvados[31], it is considered as the *"future trend"* [175] and *"lingua franca"* [144] for WMSs to achieve portability and support interoperability of workflow analyses.

CWL as a standard did not have a mechanism or a standard format for provenance capture, but rather it depended on underpinning platforms and executors to capture and track such information. As stated in *Chapter 1*, this thesis makes a contribution towards defining a standard for provenance capture that is aligned with CWL workflow enactment. Further details are presented in *Chapter 5* and *Chapter 6*.

CWL as a standard recommends, supports and encourages container-based technologies such as Docker and Singularity for resolving software dependencies. There isn't a single dedicated repository to share CWL workflows, however the research community is actively using GitHub for workflow and command-line tool specifications and Dockstore[32] for sharing these specifications as Docker images.

[31] https://arvados.org/
[32] https://dockstore.org/

2.4.3.2 Nextflow

Like CWL, Nextflow [218] is also declarative in nature with roots in the "Make-like" approach. It aims to support workflow-centric research using parallel and distributed computing resources. Nextflow includes a Domain-Specific Language (DSL) for abstract "process" declarations that can be interpreted by the executor based on a user's choice. There is no standard representation, definition or capture mechanism for provenance as yet. However, human-readable metadata[33] is available and can possibly be utilised for this purpose, if supported by the associated executors.

For software environment preservation and tools availability, Docker, Singularity and Conda support is available. For example, researchers can work in specific Conda environments where images/recipes of existing tools are readily available that address tool dependency issues. An going effort is exploring support for Research Objects for Nextflow workflows to aggregate methods, provenance and data [219, 220].

2.4.3.3 Workflow Definition Language (WDL)

WDL started as a pipeline system specific to the Broad Institute's internal workflow design and execution. WDL is accompanied with a Java-based WMS [194]. It has since become an open-source community-driven effort managed by OpenWDL incorporating continuous feedback from the research community.

WDL does not adhere to a definite standard for provenance capture and sharing. Rather, it depends on the platform implementing the workflow to capture such information. It does however provide constructs to incorporate provenance

[33] https://www.nextflow.io/docs/latest/metadata.html

components such as resource requirements in the workflow description. These can be used to extract and represent such information in a standardised form if supported by the executor/platform.

WDL is designed on the principles of portability with support for multiple platforms including local systems, HPC resources and multiple cloud environments. For software preservation, Docker support is provided to create and share Docker images for the underlying tools incorporated in given WDL workflows. There are various existing official Broad Institute Docker images[34] and also a "Method Repository" associated with the cloud platform FireCloud[35] for method and configuration sharing.

2.5 Conclusions

From comprehensive literature review, it is evident that provenance is considered a core element of any computational analysis. The lineage history of a data artefact and attribution details about the researchers involved in the analyses establishes the trust on published studies. Along with provenance traces, the availability of utilised resources enables users to verify scientific claims by reproducing the original results or by re-using systems and methods for similar analyses. The research community has come a long way from conventional lab notebooks to systematic capture and digitised representation of provenance information.

A discussed in this chapter, WMS do not share a common mechanism for provenance capture, representation or sharing. The granularity and completeness of provenance captured varies across different platforms. Despite existing

[34]https://hub.docker.com/r/broadinstitute/
[35]https://software.broadinstitute.org/firecloud/

well-established community-built provenance standards, most WMS do not yet support standard representation of captured provenance and use/re-use of this information. Those that have explored this have typically created bespoke and non-interoperable solutions. With the increasing use of containerised approaches, many WMS now support aspects of the software environment capture, but often the systematic representation and aggregation of methods as described in Section 2.2.7.2 is ignored when publishing results.

Improving the understanding of provenance with respect to bioinformatics workflows and developing a common format for provenance and associated resource representation leveraging available community-driven open source standards is a key goal of this thesis. Ideally this should be flexible and allow a range of WMS to utilise this information, i.e. a single targeted implementation would never succeed given the extensive range of WMS that now exist. We explore how this can be achieved in the following chapters.

CHAPTER 3

EMPIRICAL CASE STUDY TO UNDERSTAND PROVENANCE REQUIREMENTS

"Knowing is not enough; we must apply. Willing is not enough; we must do." –Goethe

3.1 Introduction

Given the rapid evolution of the sequencing technologies, decreasing cost of sequencing and the potential of personalised medicine to disrupt traditional approaches to clinical diagnosis and treatment of disease, it is possible that in the near future, a majority of the human population will undergo genomic characterisation in some form [178]. Indeed, the ability to digitise DNA sequences has outrun the ability to store, transmit and interpret this data. Workflow-centric research deploying computational bioinformatics workflows has been widely adopted to process this exponentially growing *-omics* data. *Chapter 2* addressed the role of computational workflows for automation of the analysis of *-omics* data. As discussed, there exist plethora of workflow definition and execution approaches, resulting in diverse heterogeneous solutions with varying degree of provenance

support. To gain understanding of the provenance status and factors influencing bioinformatics workflows provenance, we implemented and enacted a complex but widely used bioinformatics workflow, using an exemplar system from each category of workflow definition approach (described in Section 2.8).

Through implementation, we identified implicit ***assumptions*** of these approaches that ultimately result in insufficient documentation of crucial workflow requirements, leading to incomplete provenance information resulting in failed re-enactment in different contexts. In this chapter, we investigate workflow approaches to highlight these assumptions and show the issues that arise leading to limited understanding of the workflow execution lifecycle due to lack of documentation and provenance traces.

This chapter[1] explores the implementation details of the case study and identifies implicit assumptions of each category. It discusses the impact these assumptions have on the granularity and completeness of the provenance of workflow enactments. We include a set of conclusive recommendations aiming to mitigate these assumptions and guide the scientific community about the choice of workflow framework that can enable capturing fine-grained retrospective provenance that can subsequently address a key application of provenance information, *reproducibility*.

[1]Elements of this chapter are published in the following article:
Kanwal, Sehrish and **Khan, Farah Zaib** and Lonie, Andrew and Sinnott, Richard O **Investigating reproducibility and tracking provenance–A genomic workflow case study.** In: BMC bioinformatics 18.1 (2017), p. 337.
First and Second author contributed equally.

3.2 Case Study

To comprehensively understand and identify assumptions that are implicit in the approaches detailed in Section 2.8, we have implemented a complex, end-to-end variant calling workflow based on the Genome Analysis Toolkit (GATK) [221] recommended best practices. This uses three exemplars as major representatives of existing workflow definition approaches used to analyse genomics data. The GATK variant discovery workflow [222] was selected because it provides clear, community advocated step-by-step recommendations for executing variant discovery analysis with high throughput sequencing data on human germline samples. The workflow exemplars comprise:

- **Galaxy** is an example graphical user interface-based integrative framework. As noted, Galaxy is an open source, web-based platform for accessible, reproducible and transparent genomics research. It supports various degrees of workflow provenance with focus on assisting the capture and use of computational methods.
- **Cpipe** is an exemplar of a bioinformatics specific pre-built pipeline, adopted by the Melbourne Genomics Health Alliance. It uses a programmatic approach which effectively includes everything necessary for reproducing a given genomic analysis, provided the same physical or virtualised infrastructure used to run the initial analysis. It includes scripts, test data, tools, reference data and databases.
- **CWL** is an exemplar of a declarative approach to workflow definition. It enables full control for users to create and run workflows using a specification which is a standardised description of the relevant environment, command line tools and other customizable settings.

3.2.1 Datasets

The GATK variant calling workflow expects the following sequence files and supporting datasets. The datasets used in this case study are provided via a cloud container and accessible through a GitHub repository [223].

- Chromosome 21 data extracted from The Genome in a Bottle dataset NA12878 [224]. This is widely used as test data because of the pre-existing and extensive analysis done on this sample.
- The agreed variant call truth set containing the known variants obtained from NIST [225]. This variant set is used for comparative evaluation to verify the results of the workflow.
- The human reference genome (hg19) and known variant databases obtained from Broad Institute [226] including 1000G_phase1.indels.hg19.sites, Mills_and_1000G_gold_standard.indels.hg19.sites and dbsnp_138.hg19
- A .bed file specific to the Illumina TruSeq platform [227] containing the exome coordinates required for the variant calling pipeline.

3.3 Workflow Design and Enactment

This section provides an overview of the enactment process[2] of the GATK variant calling workflow using the three exemplar workflow approaches. We elaborate the assumptions implicit in each approach while dealing with various workflow features.

[2]Details of execution are published in:
https://static-content.springer.com/esm/art%3A10.1186%2Fs12859-017-1747-0/MediaObjects/12859_2017_1747_MOESM1_ESM.pdf

3.3.1 Cpipe

Cpipe belongs to the category of bioinformatics specific pre-built pipelines. It was deployed on the National eResearch Collaboration Tools and Resources (NeCTAR) research cloud [228]. The instructions on the official Cpipe GitHub page were followed to clone the repository and set up the pipeline [229].

- The cloud instance launched for executing Cpipe had 16 cores and 64GB RAM. The automated mechanism to document and convey the requirements for customised analysis is not defined for Cpipe. Rather *the pre-built pipelines presume availability of sufficient compute power to deal with data intensive steps such as sequence alignment*. To cater for the storage requirements of the pipeline, a 1000GB volume was mounted to the cloud instance. As with the compute requirement, *there is no automated mechanism for explicitly recording storage requirements*. As the genomic sequence analysis involves dealing with large input and intermediate datasets (including whole genome reference data), the pre-built pipelines *assume availability of sufficient storage capacity to deal with the data storage requirement*.

- The installation script provided with Cpipe compiled tools such as Burrows-Wheeler Aligner (BWA) [230] and downloaded databases such as Variant Effect Predictor (VEP) [231] and human reference sequence files. The pre-built pipelines connect to online resources and download and compile tools and reference datasets used in the analysis. FTP clients and SSH transfer tools are used for transferring datasets over distributed resources. Hence, *the availability of high performance networking infrastructure is assumed to transfer bulk data*.

- The base software dependencies of the underlying programming frameworks such as Java and Python were required to execute different underly-

ing tools in Cpipe. The pre-built pipelines *assume that users are responsible to solve base software dependencies for the pipeline*; otherwise the pipeline would fail to execute.

- Cpipe requires downloading and pre-processing reference datasets to generate secondary index files since the indexing step is not explicitly defined as part of the pipeline but included in a separate script. *The pre-built pipelines expect users to perform pre-processing steps and hence assume availability of input data files before execution of the pipeline*.

- Cpipe uses a copyrighted tool, ANNOVAR [232], for annotating variant calls. Pre-built pipelines deploying copyrighted or proprietary tools *assume availability of all such licensed resources are available to users*.

- Cpipe requires a specific directory structure with three sub-directories specifically named as data, design and analysis with defined placement of inputs such as FASTQ files in 'data', target .bed files in 'design' and a configuration files in the parent directory. In addition, file paths are hard-coded in the script. As the pre-built pipelines are customised to support explicit analysis requirements, *these assume availability of a specific analysis environment with a given directory structure with tools and datasets appropriately located to support seamless execution of the pipeline*.

3.3.2 Galaxy

As described in *Chapter 2*, Galaxy is a GUI-based open source WMS extensively used in bioinformatics analyses. The datasets such as FASTQ files, reference sequence files and variant databases are not embedded in the history when distributed. This section discusses implicit assumptions in a given analysis leading

to coarse-grained prospective and retrospective provenance through the Galaxy workbench.

- The Genome Virtual Laboratory (GVL) [233] was used for launching a pre-configured Galaxy instance on an OpenStack-based cloud environment. An 8-core cloud instance with 32GB RAM was launched to provision a fully configured Galaxy instance for the analysis of NA12878 Chromosome 21. Similar to pre-built pipelines, GUI-based workbenches also *assume the availability of sufficient compute power to process data*, hence are dependent on users provisioning these resources.

- A 1000GB volume was mounted to the Galaxy instance. GUI based workbenches require users to provide sufficient storage capacity to deal with the data storage, i.e. *the workflows built using these workbenches do not explicitly declare such requirements*.

- Chromosome 21 FASTQ files, known variant vcf databases and hg19 reference sequence FASTA file were uploaded to Galaxy[3]. Galaxy uses inbuilt reference files if they are not provided by users whilst other databases are expected to be provided by users. Even if a complete workflow built on such systems is published, not only is the provision of input data the users responsibility, but the workbench also *assumes the availability of supporting data (such as reference sequence and variant databases) to generate similar results*.

- During implementation, it was observed that Galaxy automatically performs certain steps without explicitly declaring them. Examples of such undocumented steps include indexing the user provided reference genome, creating index files for the BAM output file (using Picard MarkDuplicates[4]),

[3]https://github.com/FarahZKhan/GATK-CaseStudy/tree/master/Galaxy
[4]https://broadinstitute.github.io/picard/command-line-overview.html#MarkDuplicates

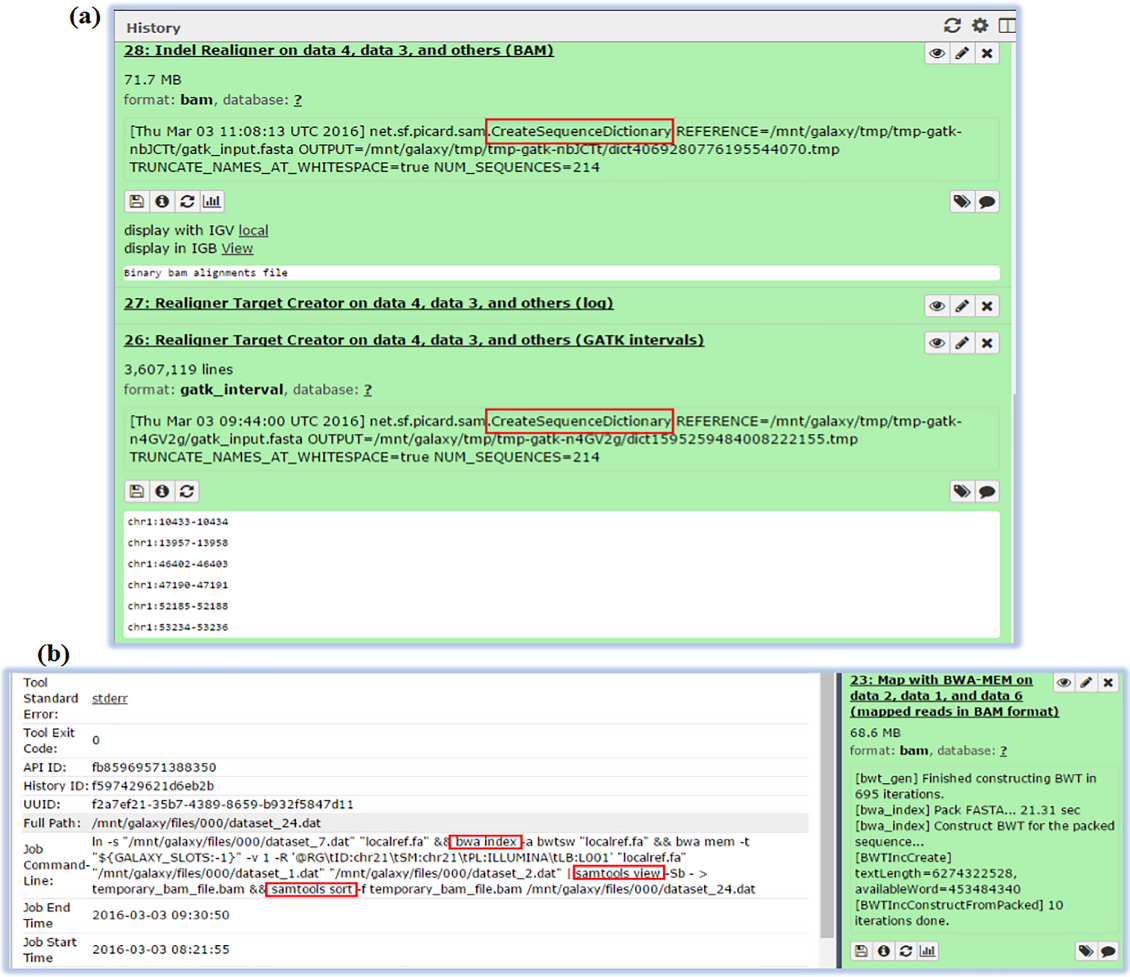

Figure 3.1: Screen shots of the Galaxy interface showing **(a)** A temporary sequence dictionary file created using CreateSequenceDictionary as part of the RealignTargetCreator and IndelRealigner steps and **(b)** "Map with BWA-MEM" step combining indexing reference data, SAM to BAM conversion and sorting of the resultant aligned (BAM) file.

generating a temporary reference sequence dictionary as part of the local realignment steps and creating a FASTA index file for GATK tools (Figure 3.1). GUI-based workbenches simplify the interface and facilitate user interaction by hiding the underlying details from the user, however this can result in an inability to replicate or reproduce the same workflow on a different platform due to incomplete provenance information that is captured.

- In Galaxy, reference sequence indexing, SAM to BAM conversion and sorting the resulting BAM file are all embedded in the alignment step and do not appear in the final workflow diagram. The visual data-flow diagram produced by such systems purport to be a complete picture of the processes carried out during a workflow execution, however this is not the case. The absence of an entire step of pre-processing, processing or post processing data from the workflow details especially from visual representation leads to incomplete workflow knowledge that impacts on its reproducibility.

- The Galaxy toolshed is populated with tools configured using XML specifications. Technical and extensive programming expertise is required to write XML configuration files for the tool versions that are not available in the toolshed. A Galaxy workflow is based on a local Galaxy toolshed, therefore a workflow created using particular tool version on one instance will fail to execute on instances with a toolshed supporting different tool versions. This renders it inflexible and a challenge to reproducibility. *Workflow developers assume uniformity of tool repositories and this may or may not actually be the case*. Recent support for container-based technologies and Bioconda recipes to handle otherwise complicated software dependencies [8] is a crucial step in this direction to overcome these issues, however at the time of our work, these approaches were still in their infancy.

3.3.3 Common Workflow Language (CWL)

CWL aims to provide a standardised approach to workflow definition by providing declarative constructs for workflow and command line tool definitions. A workflow in CWL is composed of steps where each step refers either to a command line tool (specified using CWL) or another workflow specification based on

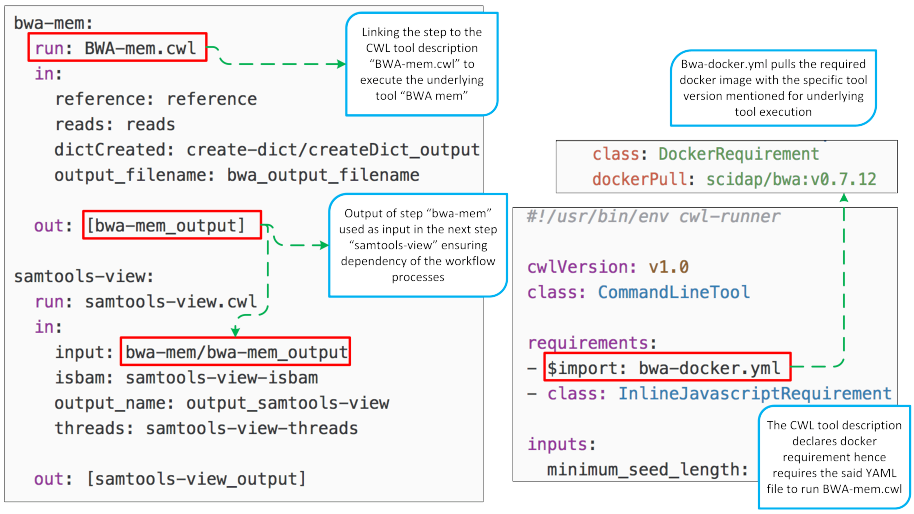

Figure 3.2: Right: A snapshot of part of a GATK workflow described using CWL. Two steps (*bwa-mem* and *samtools-view*) are shown where the former links to the tool description executing the underlying tool (BWA-mem for alignment) and provides the output used as input for samtools. Left: Snapshot of BWA-mem.cwl and the associated Docker requirements for the exact tool version used in the workflow. execution.

the concept of sub-workflows. Each step is associated with inputs that are comprised of any data artefact required for the execution of that step (see Figure 3.2). As a result of the execution of each step, outputs are produced which can become (part of) inputs for the next step(s) making the execution *data-flow* oriented. CWL is not tied to a specific operating system or platform but supported by a range of platforms such as Arvados and Toil.

- To enact the variant calling workflow, a reference implementation *cwltool* [234] was cloned and installed from source within a Python virtual environment following instructions from the GitHub repository[5].

- Working with CWL can be challenging as compared to Cpipe and Galaxy as CWL is an ongoing, constantly evolving community effort in a relatively early stage of development. Tool specifications for most of the required

[5]https://github.com/common-workflow-language/cwltool

tools for this study were not initially available. Implementing the GATK workflow in CWL required the development of a number of CWL definition files including Yet Another Markup Language (YAML) tool wrappers, JSON job files containing the input parameters and YAML test files for conformance tests. These were produced for tools from Picard Toolkit [235] (MergeSAM, SortSAM and MarkDuplicates) and GATK Toolkit (RealignTargetCreator, IndelRealigner, BaseRecalibrator, PrintReads and HaplotypeCaller) as part of the study. Any user wanting to utilise these definition files along with the workflow definition should have basic understanding of YAML and JSON. In addition, if a newer version or different tool is required for any step, the user is expected to develop the definition files which may require in depth knowledge of the underlying languages. Standardised approaches such as CWL on the one hand provide users with the freedom to declare every aspect of the workflow, but on the other hand they can *assume detailed implicit knowledge of underlying languages and standard leading to a potentially steep learning curve for users.*

- The Docker images for BWA version 0.7.12 and SAMtools version 0.1.19 were already available in Docker hub. However to work with the specific Picard toolkit (1.136) and open-source GATK-2.8 versions, a new Docker image was required. This was created and made available on the Docker hub registry for public use [236]. Using Docker images to satisfy the underlying software dependencies requires installation of Docker which again was assumed to be available on the system executing the workflow. Although CWL encourages use of container technologies, it is not a *"MUST have"* feature. Users are also able to use local installation of the required tools. This approach is not preferred however as it can lead to localised solutions that fail to execute elsewhere. In both cases, various assumptions need to be made regarding the availability of the underlying tools and their link with

tool definitions. Hence, even for standardised approaches that require the explicit declaration of every step of the workflow, they also assume the underlying software availability for enactment of the workflow, which is not always the case.

- Lastly, as genomic workflows usually involve working with large datasets, the availability of compute and storage resources is assumed to be specified explicitly by users and managed by platforms to successfully enact workflows. As a standard, CWL provides some constructs that allow users to specify this information but the actual management is left as platform-specific actions.

3.4 Features of Bioinformatics Workflow Provenance

Variant calling workflows produce genetic variation data that serves to enhance understanding of biological processes that can impact directly on the health of individuals. Therefore it is essential that complete provenance capture of data processing must be ensured to guarantee reproducibility of the research, especially given the health focus and impact on individuals. However, a generalised set of rules and recommendations to achieve this is still problematic, as workflow implementation, storage, sharing and reuse varies significantly depending on the choice of approach and platform used.

The implicit assumptions that encompass the intricate details of the workflow enactment identified in the previous section are rarely held true given the diversity of tools, environments and lack of consensus on provenance information. As a result, the workflow authors overlook sharing provenance information about the key artefacts leading to failed enactments when reproducibility of the

experiment is attempted. In this section, we identify and discuss such general features/components of provenance traces for a typical bioinformatics workflow that must be documented to support their reproducibility. We also provide recommendations to workflow developers and users for complete documentation of the process and availability of related resources required for genomic data processing workflows.

3.4.1 Workflow Enactment Environment

Provenance tracking and reproducibility go hand in hand. Provenance traces contribute to make any research process auditable and results verifiable [97]. If accompanied with the resources referenced in the provenance trace then the results may be reproducible. This section details environment-specific characteristics and their effect on the reproducibility of a given analysis.

The reproducibility of an experiment often requires replication of the precise software environment including the operating system, the base software and software dependencies and the associated configuration settings under which the original analysis was conducted. A tool or a workflow built on a specific computing platform requires the details of the exact version of the underlying base software to execute successfully. One example in this case study, was the requirement of a particular version of Java (1.8) to execute tools from the Picard toolkit used in the workflow. The absence of such information about the base software requirements such as Java or Python results in the unsuccessful execution of the workflow.

Detailed provenance information on resources such as required software versions, Docker image IDs, operating system type/version and parameter space

used for the workflow should be provided along with the workflow specifications to aid reusability and reproducibility of the workflow. Ideally workflow developers should package all associated tools when the workflow is published. Workflows should be treated as first class data objects [91] leveraging container technologies such as Docker, Singularity or LXC to package the environment and configuration information together.

Provenance traces encompassing details of the required software environment and availability of the required software via container-based images and configuration managers help to overcome complex inter-dependencies between software packages used in a workflow. The burden obviously belongs to the workflow developers but in the longer run, it would be helpful to declare and document such key information as part of workflow provenance.

3.4.1.1 Analysis Environment

In our case study, creation of an analysis environment with a particular directory and file naming convention was required by Cpipe to execute the workflow successfully [181]. Explicit requirements for specific analysis environment, e.g. hard coded paths and names embedded in source code, should be avoided to ensure portability of workflow analyses. Such details captured in the provenance trace are neither informative nor helpful as they still require a manual process on the part of the researcher re-using the shared workflow. More generally, extra responsibilities on the researcher reproducing someone elses workflow is to define the analysis environment and related parameters, e.g. providing hard coded file names, resolving absolute file paths, host names, user names and IP addresses. Workflow developers should ensure their workflows are independent of the specific analysis environment to allow their workflows to be more readily re-usable.

3.4.2 Data Availability

Input such as sequencing reads in FASTQ files and reference datasets play a major role in reproducibility of genomic workflows and ultimately achieving repeatable results. Even in the case where the user has comprehensive understanding of the workflow analysis, absence of domain-specific input data hinders the successful execution of workflows. Analysis tools usually require strict adherence to file formats (e.g. reference sequence should be a single reference sequence in FASTA format or the names and order of the contigs in the reference data used must exactly match one of the official reference canonical orderings [19]). This requires access to primary data used in the analysis. However, a major challenge in this regard, lies in the security and ethical consideration of access to and use of genomics data. The community needs to address this issue by providing secure controlled access to such sensitive genomic data. Where possible, example datasets should also be provided with a truth-set for testing the methods shared.

The size of genomic datasets can also be a problem when sharing data and providing them as inputs to workflow specifications. The download of large genomic datasets from third party online resources demands users have access to high performance networking infrastructure. In cases where it is not possible to package or share datasets with the workflow, comprehensive domain-specific annotations captured as part of provenance traces shared with the workflow can assist researchers and help them decide for example on alternative datasets that can be used for the workflow. Another possible solution is archiving big datasets in online repositories[6],[7] or data stores and just preserving the persistent identifiers as part of provenance trace to be shared with the published study. Ultimately however such large data sets needs to be made available on the system enacting

[6]http://www.data.cam.ac.uk/funders
[7]http://www.nature.com/sdata/policies/repositories

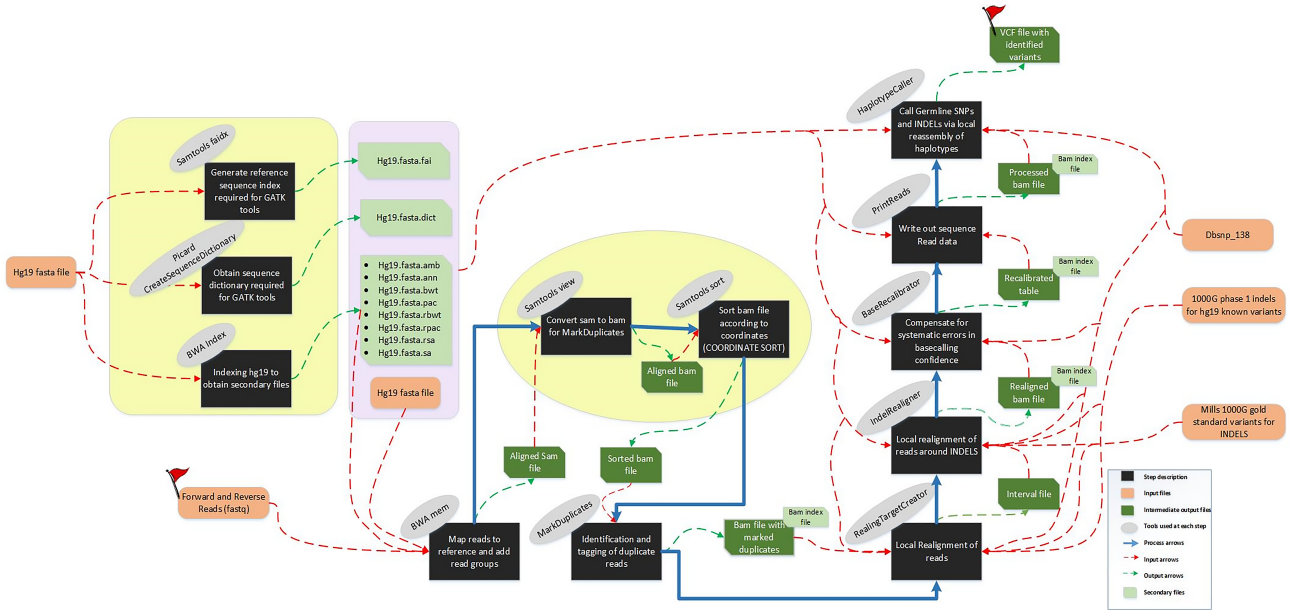

Figure 3.3: Graphical representation of the GATK workflow representing artefacts and information necessary to be captured as part of a given workflow execution. The description of the main steps is depicted in the black rectangles whereas the tools responsible to carry out the steps are shown in grey ellipses. Input and reference files (brown rounded rectangles) are shown separately and labelled by the dataset name. The primary and secondary output files (if any) are shown in dark and light green snip diagonal corner rectangles respectively. The input and output data flow for each workflow step is demonstrated through red and green dotted arrows respectively. The connection between processes in a workflow is represented by blue solid arrow. The yellow highlighted parts of the workflow are the pivotal processes not explicitly declared in Galaxy and Cpipe. The red flag highlights the main input and final outputs required for the workflow.

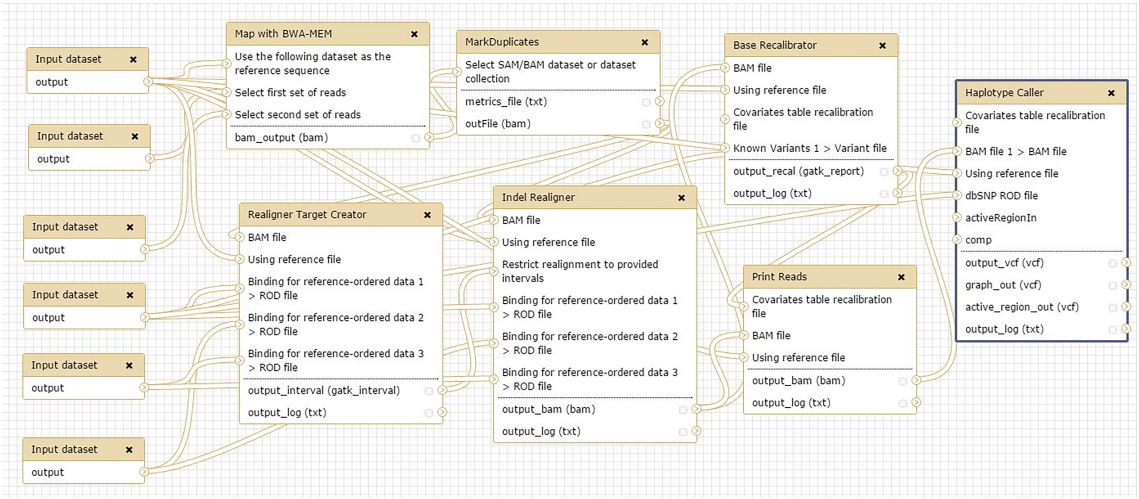

Figure 3.4: The variant calling workflow representation in Galaxy

a given workflow.

3.4.3 Abstract Representation of Workflow

As noted, during this study we observed that the Galaxy workflow graphical representation does not explicitly state the utilisation of some tools such as BWA Index, SAMtools View, SAMtools Sort, SAMtools Faidx and Picard CreateSequenceDictionary. Therefore the Galaxy workflow diagram (Figure 3.4) does not convey all information required for understanding the requirements of the workflow to be able to reproduce it on other platforms. This can simplify Galaxy users experience, but makes the workflow incompatible with other non-Galaxy platforms. Platforms making assumptions about aspects of workflows without documenting them, e.g. through graphical representations and workflow diagrams, result in incomplete prospective provenance.

The details vital to understand and reproduce a computational genomic analysis should be completely documented to ensure capture of critical provenance

information. From the experience gained from this study, we posit that the workflow developers should collectively provide abstract high-level representations of workflows documenting key information, e.g. through graphical representation of the workflow as indicated in Figure 3.3. The flowchart in the figure can be used as a model to record a high level representation of the underlying complex workflow and act as a blueprint containing all artefacts needed/used, including tools, input data, intermediate data products, supporting resources, processes and connections between the artefacts. To re-enact any workflow, users should be directed to explicitly understand and declare the requirements documented in such workflow representation.

Furthermore, the proposed representation of the variant calling workflow shown in Figure 3.3 should be preserved as a key part of prospective provenance as it contains crucial information about the data-flow and expected data artefacts needed to re-enact the workflow across the platforms. The concept of visual representation of the workflow is currently implemented in only a few GUI based workbenches [237, 6, 8]. However high-level representations often only depict an inadequate illustration of the tools and data flows as evident from Figure 3.4.

Workflows used to implement biomedical data analyses are complex [238]. This makes it difficult to understand and reproduce them. A graphical representation such as Figure 3.3 allows for the visualisation of multiple aspects of the workflow definition and implementation including the data manipulation and the interpretation. Enabling simplicity by representing complex workflows in human readable forms can significantly reduce the complexity of such analyses through improved understanding. As studies involving complex analysis tasks typically encompass human judgements, it is important that the research community works in this direction to help researchers transfer their knowledge and expertise using proposed rich and easy to create and understand representations.

Human readable descriptions along with the machine readable ones as part of prospective provenance can help identify bottlenecks in analysis and ultimately accelerate reproducibility of data-driven science.

3.4.4 Compute & Storage Requirements

Owing to the production of exceptional amounts of genomics data, a typical human exome sequence analysis would require a terabyte of storage and up to 64GB RAM of compute power for processing the data efficiently. As the computational dependencies of workflows have grown from simple batch execution models to leverage distributed and parallel processing algorithms and resources, researchers should document the amount of storage and compute power required (as an element of prospective provenance) and utilised (as an element of retrospective provenance) by a workflow to run successfully. From the three exemplar systems analysed, CWL provides constructs to document this information as *"resource requirements"* at the workflow as well as the individual step level. However, it relies on the workflow platform to utilise this information for actual resource allocation.

3.4.5 Online Resource Challenges

Genomic data analysis has grown complex with the increased involvement of customised scripts and online resources needed to carry out difficult tasks. This increases the technical knowledge required and the chance that something will break based on potentially volatile, remote resources. One of the major reasons for non-reproducibility of workflows is use of third party resources including databases, tools and/or websites [21]. Many workflows cannot be enacted be-

cause third party resources they rely on are no longer available or the software version has changed. These factors can be considered out of the control of researchers as every time an analysis is repeated, it may require/assume that the system it is being reproduced on comes pre-configured with all the workflow dependencies integrated.

Unavailability of potentially volatile online resources is an open problem that impacts on many domains. Several solutions have been proposed including use of alternative resources or keeping local copies [22]. However alternative resources might not result in the same output, hence they can be a barrier to reproducibility of results [239]. The services hosting third party resources are generally under no agreement to continuously supply such resources and/or versions of these resources. Even the most sophisticated and widely used technologies such as container-based approaches require a connection to the network and online resources (at least once) for building the required software components.

3.4.6 Proprietary & Copyrighted Resources

As noted, third party resources such as copyrighted or proprietary software and data resources should be avoided in research involving use of genomic datasets as they can result in an inability for many to access these resources due to licensing issues. The possible solutions to reproduce the research involving such tools can be through buying the software or re-implementing the licensed tools using open source solutions, which is often not realistic. To avoid this situation, the community should wherever possible, push towards *Open Science* to encourage open source software, open licensing and support for collaborative science [240]. It should be easier to communicate and access scientific knowledge. The efforts such as Centre for Open Science [241] and Journals such as *PeerJ* and *GigaScience*

are working towards encouraging such openness and reproducibility of scholarly research to accelerate scientific progress. The research presented in the later chapters of this thesis leads by example and puts emphasis on following the principles of *Open Science* and making explicit recommendations and choices of workflow approaches to encourage transparency of research.

3.5 Why Declarative Systems?

Workflows are often (typically!) dependent on the replication of complex software environments necessitating substantial technical support to reproduce the configuration settings required for the analysis. This varies depending on the different approaches taken to workflow design and execution. The assumptions followed by each approach are one of the reasons for this heterogeneity, which as noted results in inadequate documentation of workflow requirements leading to incomplete provenance information. The case study illustrates the variability in workflow implementations based on the platform selected that can impact on the capture of crucial elements of provenance currently missing from workflows. Declarative approaches like CWL make minimal assumptions about the software environment, base software dependencies, configuration settings, alteration of parameters and software versions. Such approaches aim to build flexible and customised workflows that can include intricate details of every process in a workflow. CWL provides explicit constructs to declare compute and storage level requirements, which should be encouraged/recommended as part of best practices of any workflow design. This results in the capture of essential elements of provenance and archiving of the entire software environment that can be re-established when required at a later point.

However, working with fully declarative workflow languages is not an easy

task and requires considerable time, effort and substantial technical support (in the case of this case study this was provided by the CWL community) to first learn the principles of the language and then practical coding to implement system configuration of complex genome analysis workflows. Adoption and uptake of CWL will encourage a community that can provide prompt guidance and mitigate the effects of the current steep learning curve. The most recent studies in the field of workflow-centric bioinformatics research also identify these approaches as *"Future trends"* [175, 127] that aim to improve experimental design by providing a common format and portable workflow definitions.

3.6 Conclusions

Every new discovery in science is built on existing knowledge; that is, published literature acts as a building block for new findings or discoveries. Using published literature as a base, the next level of understanding is developed and hence the cycle continues. Research is dependent on computational analysis and reproducibility not only requires an in depth understanding of science but also complete and automatic documentation of provenance encompassing details of the data, methods, tools and computational infrastructure. The challenges of large-scale genomics data demands complex computational workflow environments making comprehensive provenance documentation a non-trivial task. A key challenge is improving capture, representation and sharing of the provenance of these experiments involving complex software environments and large datasets. In this chapter we have focused on genomic workflows to understand the features of provenance effecting the understanding and reproducibility of bioinformatics-specific computational analyses.

Assumptions	Recommendations
Availability of sufficient storage and compute resources to deal with processing of big genomics data	Workflow developers should provide complete documentation of compute and storage requirements along with the workflow to achieve long-term reproducibility of scientific results.
Availability of high performance networking infrastructure to move large-scale genomics data	Considering the size and volume of genomic data, researchers reproducing any analysis should ensure that an appropriate networking structure for data transfer is on hand
The computing platform is preconfigured with the base software required by the workflow specification	Workflow developers should provide a mechanism with checkpoints to ensure compatibility of the computing platform deployed by a researcher to reproduce the original analysis
Users are responsible for ensuring access to copyrighted or proprietary tools	The broader community should encourage use of open source software and collaborative approaches and wherever possible avoiding use of copyrighted or proprietary tools
Analysis environment with particular directory structures and/or file naming conventions are set up before executing the workflow	Workflow developers should avoid hard-coding localised parameters such as file names, absolute file paths and directory names that would otherwise render their workflow dependent on a specific environment setup and configuration
Appropriate datasets are used as input to the tools incorporated in the workflow	As bioinformatics analysis tools require strict adherence to input or reference file formats, data annotations and controlled access to primary data should be included in provenance traces making it domain-specific to help make the workflow reproducible

Assumptions	Recommendations
Users must have a deep understanding of the analysis and the provided information given in the workflow	Workflow developers should provide a complete data-flow diagram serving as a blue print containing all artefacts including tools, input data, intermediate data products, supporting resources, processes and the connection between these artefacts as part of prospective provenance
Availability of specific tool versions and setting relevant parameters	Tools should either be packaged along with the workflow, containerised or made available via public (accessible) repositories to ensure that the same versions and parameter settings are used in a given analysis to support flexible and customizable workflows.
Users need to have proficient knowledge of the specific reference implementation	This factor might be considered out of control of the workflow developers but detailed documentation of the underlying framework used and associated community support can help overcome associated learning curves

Table 3.1: Summary of assumptions and corresponding recommendations regarding workflow environments

This chapter provided an overview of the implementation of the genomic variant calling workflow using three exemplar workflow definition and implementation approaches. In implementing this workflow, a set of assumptions common to the three approaches was identified. We also identified unique challenges and assumptions particular to each platform through this practical enactment of the workflow. These assumptions often considered needless to be stated by the workflow authors result in lack of necessary details when an analysis is

published. This leads to heterogeneous and incomplete documentation of retrospective and prospective provenance, even for the same workflow on the same platform. This is exacerbated when a different workflow platform is considered for re-enactment, hence have a direct impact on reproducibility.

We conclude this chapter with a set of recommendations (Table 3.1) to be considered when designing or re-using a workflow specifically in the genomics domain to address the aforementioned assumptions. We posit that adhering to the proposed recommendations along with an explicit declaration of a standardised workflow specification will lead to fine-grained retrospective provenance capture of computational genomic analysis. Ensuring reproducibility of a given analysis ultimately depends on the efforts from researchers to communicate their analysis in a comprehensive, standardised and understandable manner. The next chapter further elaborates the best-practice recommendations associated with workflow-centric studies and defines a hierarchical representation of provenance framework to support homogeneous workflow and resource communication.

CHAPTER 4

LEVELS OF PROVENANCE AND RESOURCE SHARING

"The best minds would be led to contribute to further progress, each one according to his bent and ability, in the necessary experiments, and would communicate to the public whatever they learned, so that one man might begin where another left off; and thus, in the combined lifetimes and labours of many, much more progress would be made by all together than any one could make by himself." –René Descartes

4.1 Introduction

In *Chapter 3*, we focused on three broad categories of workflow definition and execution approaches (Section 2.8) using exemplar WMSs (Section 3.2) representing each approach to implement a widely used complex genomic workflow. Through this empirical analysis, we identified implicit and explicit *assumptions* made by each approach that result in incomplete and coarse-grained provenance documentation. We concluded that declarative approaches to workflow definition should be adopted for specifying workflows, since these approaches make the least assumptions about the artefacts involved in the workflow design and enactment. Workflow specifications defined in a declarative fashion using standards

like Common Workflow Language (CWL) aid in the complete capture of prospective and retrospective provenance.

The analysis in *Chapter 3* illustrated the key provenance features of bioinformatics workflows (Section 3.4) that must be documented for finer-grained provenance capture. In this chapter, we present a pragmatic analysis of existing research focused on improving the design, implementation and publication of workflow-centric analyses. The aim of this chapter is to further characterise the fundamental resources identified from the empirical analysis in *Chapter 3* and the best practice recommendations from related literature discussed in this chapter by defining a platform-agnostic hierarchical framework of provenance. This generic framework can be well applied to any workflow definition approach and systematically illustrates the understanding and implications of the identified key factors on completeness of provenance documentation. It is expected that collectively achieving all the levels of this framework will result in re-executable and comprehensive workflows equipped with detailed provenance information.

4.2 Best Practice Recommendations

Best practice recommendations for workflow-based analyses are generally derived from the experimental and pragmatic experiences of multiple research groups working on a similar research topic; these recommendations are typically published and gain community agreement and support as standard practices in a given domain. In the bioinformatics domain however, a single set of accepted standards for workflow definition or sharing has not been agreed upon for workflow-oriented analyses. Indeed as noted in *Chapter 3*, there is substantial heterogeneity in the existing approaches for workflow definition and implementation. There exist various studies working on the same problem i.e. improving the work-

flow design, data handing, provenance recording and sharing of these resources. However, similar to the variety present in workflow definition approaches, the recommendations and best practices are also distributed widely and to the best of our knowledge lack a centralised view consolidating all such experiences and propositions.

In this section[1] we consider and devise a comprehensive compilation of the common recommendations, best practices and standard approaches for workflow design and workflow-centric analyses sharing that focus specifically on improving the reliability, understandability and reproducibility of published findings. This compilation (shown in Table 4.1) brings together experiences of the scientific community on one platform that is of benefit to the research conducted in this thesis and will advance the related studies in future. Through the pragmatic analysis of these recommendations, we identify fundamental artefacts and practices crucial for capture of comprehensive provenance of workflows and support the transparent sharing and reproducibility of workflow-centric studies. It is to be noted that this compilation is not exhaustive and other requirements need also be considered, e.g. provenance analytics, data security and provenance storage are beyond the scope of this thesis.

Req.no	**Recommendation**
R1 parameters	All parameters used for the software used in a given workflow (including default values of parameters used) should be saved and shared [19, 89, 242, 243].

[1]Elements of this chapter are submitted in the following article:
Khan, Farah Zaib and Soiland-Reyes, Stian and Sinnott, Richard O. and Lonie, Andrew and Goble, Carole and Crusoe, Michael R. **"Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv".** In: GigaScience (10.5281/zenodo.1966881)

Req.no	Recommendation
R2 automate	Manual processing of data should be avoided and if *shims* [244] are incorporated for any processing, these should be a part of the workflow to fully automate the computational process [19, 243].
R3 intermediate	Intermediate results should be published with a given analysis where possible [89, 242, 243].
R4 sw-version	Exact software versions used for an analysis should be recorded [19, 243].
R5 data-version	If public data (reference data, variant databases) is used, then it is necessary to document and share the versions used [180, 245, 19, 243].
R6 annotation	Annotation tools such as user contributed tags and versions should be assigned to workflows and shared when publishing the workflows and their associated results [246].
R7 described	Workflows should be well-described, annotated and offer associated metadata [29, 91, 242, 246, 247] .
R8 identifier	Stable identifiers should be used and stored for all artefacts including the workflow, datasets and the software components [246, 247, 144].
R9 environment	The details of the computational environment used in an analysis should be shared alongside the analysis [29, 245, 247].
R10 workflow	Workflow specifications/descriptions used in the analysis should be disseminated when publishing the analysis [29, 89, 242, 247, 248].
R11 software	The software utilised in an analysis should be aggregated and shared when publishing methods and results [29, 245, 247, 248, 242].
R12 raw-data	The raw data used in a given analysis should be available, e.g. in publicly accessible repositories [29, 89, 242, 247, 248].

Req.no	Recommendation
R13 attribution	Attributions and citations related to third party data resources and software systems used should be stored and shared [242, 248].
R14 provenance	Workflows should be preserved along with provenance traces of the enactments that result in the published data and results [29, 91, 242, 243, 248].
R15 diagram	High-level data-flow diagrams or sketches of the computational analysis using workflows should be provided [245, 89, 249].
R16 open-source	Open source licensing of methods, software, code, workflows and data should be adopted wherever possible and proprietary resources deprecated [245, 89, 243, 247, 248, 250].
R17 format	Data, code and all workflow steps should be shared in a format that others can easily understand preferably, ideally in a system neutral language [29, 89, 250].
R18 executable	Easy execution of workflows should be supported without major changes to the underlying environment [180, 249].
R19 resource-use	Information about the compute and storage resources should be captured, stored and shared as part of the workflow [245].
R20 example	Example input and output data should be preserved and published along with the workflow-based analysis if primary data is unavailable [29, 22, 249].

Table 4.1: Summary of recommendations from various studies covering best practices on reproducibility, accessibility, interoperability and portability of workflows

Sharing information about the *process* followed for derivation of a given data

artefact is of paramount importance given the distributed and collaborative nature of bioinformatics research. The best practice recommendations from various research groups (Table 4.1) provide insights into resources that should be published alongside the results to describe the automated *process* followed in workflow-centric analyses. Adoption of these practices when publishing artefacts related to a new research can contribute to better computational experiment sharing. In the next section we discuss the implications of these best practices and the impact of their presence/absence on the different provenance applications discussed in Section 2.2.3.

4.3 Resources Supporting Provenance Applications

This section discusses the essential resources and design choices that impact on supporting provenance documentation in workflow-centric analyses. These resources are identified on the basis of experiences gained from empirical analyses of workflow definition approaches (Section 3.2) and pragmatic analysis of related workflow-centric studies. We have discussed related concepts together to avoid redundancy in discussion.

4.3.1 Parameter Settings

In bioinformatics analyses such as presented in *Chapter 3*, the quality and accuracy of results of a given workflow enactment has a strong correlation with the choice of parameters used for software executed as part of the workflow [251]. Emphasis on publishing not only the description of the software, e.g. the version, but also the configuration files and parameter settings as suggested by ***R1-***

parameters greatly impacts on the reproducibility of results [19]. Garijo *et al.* [89] discuss the impact of the absence of such configuration settings on the reproducibility of studies and emphasise on sharing of the configuration parameters for each software used in an analysis.

Several tools use elements of randomness, e.g. random seeds. One example is the widely-used alignment tool BWA-mem, which produces different results, if the number of threads varies based on the availability of compute resources. Unintentional randomness in such analyses will result in slightly different results on every execution [243] even using the same input and computational methods. Sandve *et al.* [243] identify that it is necessary to document such randomness factors, and if possible share the actual random numbers used in a given published analysis. Moreover, it is common practice for different tools to have different default values of parameters between versions. Therefore statements like *"Tool X was executed with the default settings"* does not provide sufficient information required to reproduce the scientific findings. Thus, sharing such settings along with published results will aid in the *reproducibility* of a given analysis.

4.3.2 Automating the Whole Process

A key advantage of utilising workflows for data-intensive computational analyses is *Automation* of the whole process in addition to Scaling, Adaptation and Provenance support (ASAP) [3]. In order to benefit from the automation offered by use of workflows, it is vital to avoid any manual interventions, e.g. for pre/post processing of data as suggested by ***R2-automate***. However, ad hoc data manipulations are often performed to achieve format compatibility between two tools, e.g. using Shims [252]. A shim refers to an adaptor or intermediary step that is required to resolve format incompatibility issues between two workflow

steps. Shim steps employ tools to convert the output of the previous step into an acceptable format for the next step in a workflow. Examples of shims from the variant calling workflow implemented in *Chapter 3* is conversion of SAM file formats produced by BWA-mem to BAM file format and subsequent sorting using SAMtools to make the alignment file analysis ready for the next step and use of Picard MarkDuplicates. These steps were incorporated as part of the workflow and explicitly declared in case of the CWL workflow definition. Sandve *et al.* [243] and Nekrutenko *et al.* [19] also suggest utilising available format converter tools, e.g. SAMtools, to achieve these tasks and make shims part of the actual workflow and hence documented as part of the workflow. This encourages *transparency* of the research process and facilitates the *reproducibility* of shared workflows.

4.3.3 Data Sharing

R3-intermediate, ***R12-raw-data*** and ***R20-example*** of Table 4.1 provide recommendations focused on sharing different types of data artefacts. These artefacts are unique in their utility with respect to provenance but they also share some common aspects and support different applications of provenance.

4.3.3.1 Raw Input Data

Input datasets utilised in an analysis are recommended to be shared in publicly accessible repositories. Availability of data used to generate published results support verification of claims by researchers by facilitating direct *reproducibility* of the shared analysis. Stodden *et al.* [247] characterise input data as the minimum component that enables independent regeneration of results. In the variant calling workflow implemented in Section 3.2, FASTQ read files and auxiliary

datasets such as the human reference genome sequence and variant databases are examples of input datasets. In the case study, these were shared via cloud object store[2].

4.3.3.2 Intermediate Results

It is quite possible that, in the case of third party data resources, despite authors providing information about the input datasets, the workflow fails to re-enact at a given time due to changes in the third party resources. This was demonstrated by Garijo *et al.* [89]. In such cases, intermediate datasets from the original analysis can be utilised to *understand* and examine the individual as well as collective computational processes that were performed to generate given results. This aids in the validation of published results and improves the overall research *transparency*. Intermediate datasets can also be used for comparison of results at each individual workflow step, e.g. by re-enacting only that step with different software or configuration settings [243]. A caveat to this approach however, is the storage overheads especially in the case of data-intensive genomic workflows. Sharing intermediate data can result in significantly greater demands for data storage. In case of storage limitation, the checksums of the intermediate data should be shared at the very least that can be used for content comparison for the individual steps of workflows.

4.3.3.3 Example Input & Sample Output

Given the nature of genomic analyses, a critical aspect to consider is involvement of human subject data as input. This raises obvious security and privacy con-

[2]https://github.com/FarahZKhan/GATK-CaseStudy/tree/master/Data

cerns. The methods however can still be shared and example input data may be provided to gain insights about the anticipated inputs and outputs. Such datasets should enable test runs of a shared analysis in the absence of the actual input data to verify the methods that are being shared as part of the workflow. Example data also contributes to *understanding* of the workflow purpose [249].

4.3.4 Workflow Specifications & Abstract Representation

R10-workflow and ***R15-diagram*** of Table 4.1 provide recommendations related to resources that can augment prospective provenance capture. As described in Section 2.2.2.1, a well-defined and well-documented workflow is a source of prospective provenance since users can infer from the workflow graph the general method used for the analysis and hence the results. Another application of such specifications is re-usability of published methods for new research with different datasets [29]. Sharing the specifications of the workflow ensures the *transparency* of the methods employed to generate the results. This helps to establish *trust* of the actual results.

In addition to workflow specification files, high-level data-flow diagrams or sketches are also recommended to be provided with a computational analysis. As described in Section 3.4.3, the workflows in biomedical data analyses have grown increasingly complex [238] and it is now challenging to understand and reproduce them. Abstract representation of workflows shared as part of prospective provenance provides a human-readable view of the workflow and improves *understandability* by giving an overview of the major steps [89], the data types and expected compute resources.

4.3.5 Attributions & Provenance Trace

R13-attribution and ***R14-provenance*** of Table 4.1 provide recommendations emphasising on sharing attribution details of third party resources and software systems and provenance trace of a given workflow enactment. Due to software identification systems, e.g. persistent Digital Object Identifiers (DOI) and open-access repositories, direct software citation and attribution is now readily available for given scientific analyses and associated publications. This practice should also be followed in the case of workflow studies coupling different third party resources such as software and data. The scientific etiquette of attribution and citation of digital scholarly objects including data, workflows and software resources encourages fellow researchers to share their methods and datasets alongside their publications. The Software Sustainability Institute provides detailed recommendations[3] to users and software providers in how to acknowledge software and the research in which it is cited. Similar initiatives for workflow accreditation and citation should be developed to incentivise the practice of workflow sharing.

The provenance trace of a given workflow enactment is referred to as *retrospective provenance* (described in Section 2.2.2.2). Such information is recommended to be captured and shared and has been identified by numerous studies as shown in Table 4.1. Expectations of WMSs should reasonably include the ability to enable recording and exporting provenance traces in a machine-readable format for subsequent querying and mining to extract information about the artefacts involved in a given analysis as discussed in Section 2.2.6. Attribution details allow to answer key queries such as "Who authored the workflow?" or "Who ran the analysis". These should also be part of the shared provenance trace along with derivation history of a given data artefact.

[3]https://www.software.ac.uk/how-cite-software

4.3.6 Software Environment

Recommendations ***R9-environment***, ***R11-software*** and ***R18-executable*** are related to the external software environment and associated resources required for a workflow. ***R18-executable*** emphasises the promotion of workflow execution without requiring changes to the underlying software environment. Integrative GUI-based WMS (Section 2.3.2.2) are pre-configured with modular tools and often auxiliary data, e.g. human reference sequences and variant databases, hence require the least effort in terms of configuration management. However, with the increasing number and diversity of package managers and configuration tools now available, it is often necessary to resolve software dependencies automatically at the platform level for easy re-enactment of a given workflow without being tied to a particular WMS avoiding vendor lock-in.

R9-environment recommends sharing of extensive details of the computational environment used in a given analysis including the software and hardware resources as suggested by ***R19-resource-use*** (detailed in Section 3.4.4). ***R11-software*** identifies that in addition to sharing details of the software used, including the actual software in the resources shared with the published analysis should be adopted to avoid dependencies on third party (external) web services that may or not be available at a given time. Sandve *et al.* [243] suggest archiving the exact environment used in the original analysis by storing a virtual machine image of the operating system and program, however this does not capture external interactions/software. Another solution that does not require manual local installation and configuration of the aggregated software is by leveraging light-weight container-based technologies (as described in Section 2.2.7.1). These aid in resolving software dependencies that allow workflows to be reproduced. This approach leverages open access, software repositories such as Docker Hub in case of Docker images. The approach assumes installation of "Docker daemons" when

enacting Docker-based workflows.

4.3.7 Versions & Persistent Identifiers

The exact version of a software tool (***R4-sw-version***) can have a crucial impact on the enactment (or re-enactment) of computational analyses. A single software package/tool can differ in performance and output based on its version as well as variance in input/output formats, default parameter values and dependencies [243]. This can result in either different outputs produced with the same input, or unsuccessful/failed enactments. Documenting the name of the software such as "Genome Analysis Toolkit" or "BWA-mem" without version information is therefore of limited use, especially in the case when the exact version is not archived. Similarly, genomic data analyses often require public datasets such as the human reference genome, variant databases and interval lists. The versions of these datasets (***R5-data-version***) are as important for *reproducibility* as versions of the software. Indeed, Nekrutenko & Taylor [19] surveyed 50 papers that perform read mapping using BWA and found that more than half of the surveyed papers provide neither BWA version information nor the human reference genome versions. This rendered these studies unverifiable and non-reproducible.

Using a version control system such as Git for tracking workflow evolution and documenting which workflow version is used for actual results generation is also recommended. Workflow-centric studies are often exploratory in nature where researchers use "trial & error" methods to test different software systems with various parameter settings for different tasks. Providing such information facilitates sharing not only the resources, but the experiences that can provide guidance to researchers and improve the *understandability* of the research process as a whole by communicating the settings tested and paths explored.

Versions of digital scholarly objects should be associated with long-lived (persistent) identifiers (***R8-identifier***) to improve their findability and accessibility. Stodden *et al.* [247] suggests including the software version and an associated unique identifier for software citation. In the context of digital artefacts, unique identifiers are often referred to as *"persistent identifiers"* [253]. These identifiers provide a standardised mechanism for accessing a digital object and its associated metadata. Scientific domains lacking a common identifier result in numerous identifier schemes with varying degrees of stability resulting in *"reference rot"* [254]. Therefore, the choice of identifier scheme is crucial to ensure the availability of shared resources for workflow-centric analyses. Examples of some frequently used identifiers for digital objects include Digital Object Identifiers[4] (DOI), CrossRefs[5], Perma Links[6] and DataCite[7]. Any/all such identifiers should be included in the provenance information of the workflow and its resources ensuring overall accreditation and *"attribution"* of the software, data and workflow elements.

4.3.8 Metadata & Annotations

Metadata and annotations in the context of computational analyses are important for discovery, interpretation and re-use of associated digital artefacts [162]. (***R6-annotation***) suggests harnessing user-contributed tags and text descriptions to add contextual information to improve the accessibility of workflows and their associated resources. These tags can convey general as well as domain-specific information. For example, a workflow step may accept a *FASTQ file format* to perform *sequence alignment*. Galaxy and WINGS [113] are example WMSs that provide functionality to allow adding user-defined "named" tags to the data artefacts

[4]http://www.doi.org/
[5](http://www.crossref.org
[6]https://perma.cc/
[7]https://www.datacite.org/

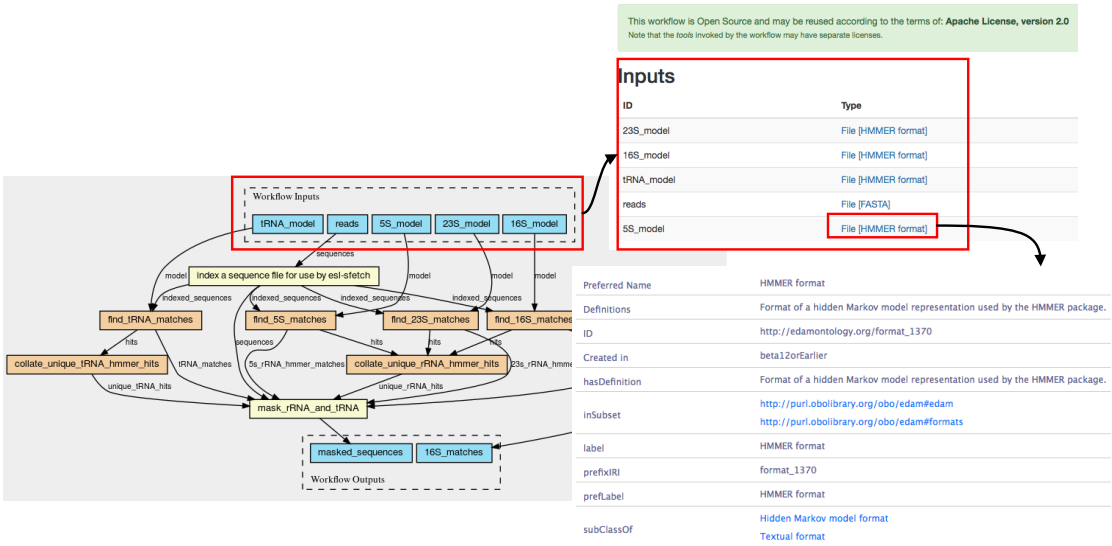

Figure 4.1: EDAM ontology utilised in a CWL workflow to declare expected input file formats

and support their propagation through the workflow lifecycle. This aids in the tracking of the data derivation history. However, manual annotations using user-defined tags without any controlled vocabulary results in heterogeneous and/or ad hoc representation of the metadata. Therefore, standardised vocabularies are preferred to record the metadata associated with data and software artefacts employed in a given workflow enactment.

It is possible to standardise the semantic annotations of bioinformatics resources utilised in a workflow by using richer and more comprehensive ontologies such as EDAM [255] and Software Ontology (SWO) [256]. These support well-known concepts related to data and software. These ontologies are comprised of well-defined modules that provide coarse-grained as well as fine-grained insights about the required input and output data format specifications, associated identifiers, precise function of the software tools, licensing information, underlying algorithm and the versions of the tools used. An example of a CWL workflow [257] using EDAM namespaces to specify the input file format is shown

in Figure 4.1. Prospective and retrospective provenance represented using a domain-neutral model as described in Section 2.2.4 can be augmented with domain-specific tags, descriptions and ontologies to improve the *understandability* of the hypothesis, choices of digital artefacts (data & methods) employed, applicability and the result interpretation of any shared workflow.

4.3.9 Open Source Licensing

In *Chapter 3*, we identified the impact of incorporating the copyrighted tool, ANNOVAR in the Cpipe workflow (Section 3.3.1), concluding that the distribution and complete sharing of the analysis methods was hindered. ***R16-open-source*** recommends adopting open source licensing as a preferred practice for publication of methods, data, software and workflows to promote the re-usability of digital artefacts. This has direct impacts on the transparency of published results and seamless (re)distribution of scientific research. The term **"Open"** [258] in the context of scientific knowledge is defined as *"anyone can freely access, use, modify and share for any purpose –subject, at most, to measures that preserve provenance and openness"*.

The three following cornerstones were identified in [258] for open scientific processes and applying as well as promoting open science practices while carrying out a given analysis:

- *open copyright* should be supported by using open source licences such as Apache-2.0 [259], MIT [260] or Creative Commons [261] for the developed methods (including tools and workflows) and generated results in a computational analysis.
- *use of open technologies* should be adopted by promoting use of open source

artefacts such as data sources and software instead of using copyrighted resources.

- *cultural openness* should be supported by embracing collaborative research for exchange and development of scientific knowledge as well as incentivising researchers by giving deserved scholarly accreditation and recognition.

Building upon existing scientific knowledge to advance research is an increasingly important expectation. This is not possible if the existing research is shared using a licence with restrictive settings thus hindering its distribution, extension or re-use. Hence, the choice of licensing, the licence of the employed tools, technologies and the associated research culture should be considered carefully when designing and publishing new work and establishing collaborations.

4.4 Levels of Provenance

Changing the research culture to embrace more digital openness and to encourage frequent sharing to go beyond the traditional methods of publication requires continuous practice and if possible formal education [262]. The sharing of *"all artefacts"* from a computational experiment or in other words following all the recommendations and best practices arbitrarily without any informed guidance is a demanding task. It requires consolidated understanding of the impact of the many different artefacts involved in that analysis. This places extra efforts on workflow designers, (re)-users, authors, reviewers and expectations on the community as a whole. In the previous section we explored the impact of crucial digital artefacts on different aspects of provenance. Given the numerous WMS and each system dealing with the provenance documentation, representation and sharing of these artefacts differently, the granularity of provenance information

preserved during each workflow enactment will vary for each workflow definition approach and even the WMS belonging to the same category. Therefore devising one universal but technology-specific solution for provenance capture and the related resource sharing is impossible. There is an urgent need of a generic framework of provenance that all WMSs can benefit from and conform to without additional technical overheads.

In this section we present such provenance framework with hierarchical levels of provenance to create awareness of the provenance requirements that need to be fulfilled for different provenance applications. The purpose of this framework is threefold. First, because of its generic nature it brings the uniformity in the provenance granularity across various WMS belonging to different workflow definition approaches. Second, it provides a comprehensive and well-defined guideline that can be used by the researchers to conduct principled analysis of the provenance of any published study. Third, due to its hierarchical nature, the framework can be leveraged by the workflow authors to progress incrementally towards the most transparent workflow-centric analysis. Overall, this framework will help achieve a uniformity of provenance and resource sharing with a given workflow-centric analysis guaranteed to fulfil the respective provenance applications.

In the following subsections, we introduce the individual hierarchical levels of the provenance documentation classifying the recommendations (Section 4.2) and elicited resources described in Section 4.3. These are represented in Figure 4.2, where the uppermost level exhibits comprehensive, reproducible, understandable and provenance-rich computational experiment sharing. Each level builds upon the lower levels and can support different use-cases. Thus whilst it would be highly desirable to support complete provenance information in all scenarios, this may not always be possible for technical or pragmatic reasons. The

levels of the devised framework are ordered from low provenance granularity to higher degrees of information specificity. In brief, **Level 0** is unstructured information about the overall workflow enactment, **Level 1** adds structured retrospective provenance, access to primary data and executable workflows, **Level 2** enhances the white-box provenance for individual steps, and **Level 3** adds domain-specific annotations for improved understanding.

The following sub-sections contain the links of each level to the requirements stated in Table 4.1 that these levels should satisfy. We refer to the authors of the original analysis as *primary authors* and any researcher utilising the shared analysis as *secondary users*. *Primary authors* can potentially be workflow developers. In principle, *primary authors* should declare the provenance level their analysis achieves when publishing it that will provide guidance to *secondary users* and set expectations for them.

4.4.1 *Level 0 –Trust, Prospective Provenance & Reuse*

Level 0, the base of the provenance framework represents the *bare minimum* that researchers can share without changing their study design or requiring any additional infrastructure or technologies. The definition of *Level 0* aligns with the concept of *"open source code"* associated with publishing software where source code is available openly for other experts to inspect, modify, enhance and re-use. In the case of workflow artefacts, the term source code can be directly associated with workflow and tool specifications. Overall, this level determines the degree of trustworthiness that can be placed on an analysis given publicly accessible resources.

To achieve *Level 0* provenance, *primary authors* should share the workflow

Figure 4.2: Levels of Provenance and their Associated Implications

specifications, input configuration parameters, raw log files and output data associated with a given workflow enactment through any open-access repository. Collectively, these resources provide the minimum information to be shared without enforcing any extra effort by *primary authors*. The artefacts shared at this level only require uploading to a repository without necessarily providing any supporting metadata. The configuration files comprising parameter settings and raw logs with the actual commands executed for each step of the workflow can be utilised by *secondary users* in debugging the workflow or workflow steps. This includes assessing the impact of certain parameters and interpretation of the corresponding results.

Workflow definitions based on *Level 0* can be re-purposed for other analyses by *secondary users*. As discussed in Section 2.2.2.1, a well-written scientific workflow and its graphical representation is itself a source of prospective provenance giving users an overview of the general method used for the production of results and characteristics of the input/output data [53]. A well-described workflow specification indirectly provides prospective provenance without aiming for it. In addition to the textual workflow specification, some WMSs also provide a graphical representation. If such a representation is available, it should also be shared as part of prospective provenance, hence fulfilling ***R15-diagram***. At this level, *primary authors* are not expected to do great deal of extra work with regards to metadata or annotation provision, although these may be added automatically by the WMS.

At this level, re-running the workflow "as-is" using only the workflow specification and configuration files is likely to require considerable extra effort by *secondary users*. This may include requesting *primary authors* to share comprehensive information about the software environment to tackle software dependency issues. However, the shared workflow/tool specifications can be *re-purposed* and

re-used [263] in subsequent analyses through varying data/configuration subject to the *secondary user's* domain-specific expertise. Ideally experts in the field who are familiar with the domain of research as well as the workflow definition approach can exploit such information for effective re-use and re-purposing. As open access journals require availability of methods and data, many published studies now share workflow specifications, input data, and the results thereby achieving *Level 0* provenance. This satisfies ***R1-parameters*** and ***R10-workflow*** automatically, hence provides coarse-grained provenance.

4.4.2 *Level 1 –Retrospective Provenance & Reproducibility*

The framework is hierarchical in terms of granularity, therefore moving up the pyramid requires finer-grained information sharing by *primary authors* to facilitate better use and enable reproducibility of the shared artefacts. At *Level 1*, *primary authors* are required to provide retrospective provenance of the workflow enactment. This includes sharing information about the data derivation process such as "what was executed", "when was it executed", "who performed the analysis", "what was used" and "what was produced". Retrospective provenance must be provided in a structured machine-readable representation that can be inspected and queried to extract answers to questions such as the above. Provision of retrospective provenance can directly satisfy ***R14-provenance*** and documentation of the exact versions of digital artefacts as part of provenance can also cater for recommendations ***R4-sw-version*** and ***R5-data-version***.

In addition to retrospective provenance, the reproducibility of analysis should also be supported by *primary authors* at this level. This requires fulfilment of recommendations ***R9-environment***, ***R11-software*** and ***R18-executable***. This can include packaging and sharing the software environment along with the workflow

utilised in a given analysis or by providing comprehensive information to guide the *secondary user* in the process of establishing the required software environment. The former is a recommended approach as it requires minimal effort and time consumption by *secondary users*, hence it increases the likelihood of reproducibility. Software packaging and configuration technologies as described in Section 2.2.7.1 should be exploited to provision the precise software environment alongside the published results.

The *primary authors* should also provide access to the input data fulfilling recommendation ***R12-raw-data***. This data can either be used to re-enact the published methods or for use in a different analysis, e.g. for performance comparison of different tools. To make a workflow re-enactable, local references such as hard-coded local paths and file names to resources should be avoided and *primary authors* should provide content-addressable data artefacts. Provision of input data coupled with containerised software environments and retrospective provenance aids verification of published results. Intermediate data artefacts should also be provided at this level to allow inspection of individual steps without re-running the full workflow, thereby addressing recommendation ***R3-intermediate***.

While software and data can be digitally captured, the hardware and infrastructure requirements also need to be captured to fulfil ***R19-resource-use*** and provide a priori estimation of the computational cost and capability needed for re-running or reusing a workflow. This kind of information can naturally vary widely with runtime environments, architectures and data sizes [264], as well as rapidly becoming outdated as hardware and cloud offerings evolve. Nevertheless a snapshot of the workflow's overall execution resource usage for an actual run can be beneficial to give a broad overview of the requirements, and can facilitate cost-efficient re-computation by taking advantage of spot-pricing for cloud resources [265].

4.4.3 *Level 2 –Towards White-box Enactment*

Modular design of scientific workflows by separating related tasks into "sub-workflows" or "nested workflows" is a common practice [144]. Such modules can be incorporated and used in other workflows or be assigned to appropriate compute and storage resources in the case of distributed computing environments [266]. Modular structures promote understanding and hence re-usability of workflows as researchers are inclined to use these modules instead of the workflow as whole for their own computational experiments [144]. An example of a sub-workflow is mandatory "pre-processing" [267]. This is needed for the Genome Analysis ToolKit (GATK) best practice pipelines which is used for genomic variant calling. These steps can be separated into a workflow that can be used before any variant calling pipeline, be it somatic or germline.

At *Level 1*, retrospective provenance is coarse-grained and as such, there is no distinction between workflows and their sub-workflows. Rather sub-workflows are treated as black boxes resulting in a form of *black-box* provenance typifying many scientific workflows (as discussed in Section 2.2.1.2). This phenomenon is prevalent in many graphical user interface-based platforms that provide levels of abstraction/obscurity to the actual tasks being implemented. As demonstrated in *Chapter 3*, declarative approaches to workflow definition result in transparent workflows with the least number of assumptions resulting in *grey-box* granularity of provenance. To further support research transparency, *primary authors* should share retrospective provenance traces for each workflow including the nested/sub-workflows and provide details of the workflow enactment as explicitly as possible to offer finer-grained, *white-box* provenance. Such provenance traces can deliver value by supporting *partial re-runs*, i.e. by only using some of the workflow components. At this level, the *secondary users* should be able to inspect and re-run the targeted components of a given workflow, e.g. a sin-

gle step or specific sub-workflow without necessarily having to re-enact the full workflow.

Primary authors should also include persistent identifiers to identify workflows and their individual steps as produced in the provenance traces. These identifiers can aid in the subsequent analysis of workflow traces. As discussed in Section 4.3.7, such identifiers also encourage *primary authors* to consider the longevity of shared resources, hence supporting recommendation ***R8-identifier***.

Improving ***R19-resource-use*** would for *Level 2* include resource usage per task execution. Along with execution times this can be useful information to identify bottlenecks in a workflow and for more complex calculations in cost optimisation models [268]. At this level resource usage data will however also become more noisy and highly variant on scheduling decisions by the workflow engine, e.g. sensitivity to cloud instance reuse or co-use for multiple tasks, or variation in data transfers between tasks on different instances. Thus *Level 2* resource usage information should be further processed with statistical models for it to be meaningful for a user.

4.4.4 *Level 3 –Understandability & Specificity*

Levels 0-2 are generic and domain-neutral, and can apply to any scientific workflow. However, domain-specific annotations about digital artefacts involved in a given analysis plays an important role in the understanding and exploitation of provenance information, e.g. for meaningful queries to extract information related to the domain under consideration [269]. An example of such bioinformatics resources annotations is specific file formats for datasets in variant calling workflow such as FASTA, FASTQ and BAM that represented and iden-

tified using well-defined EDAM bioinformatics operators[8] *"format_1929"*, *"format_1930"* and *"format_2572"* respectively. In a workflow specification, using such controlled vocabularies to convey information about the required input datasets or expected outputs help understand the requirements better. Hence, domain specific metadata such as file formats, user-defined tags and other annotations can augment generic retrospective provenance and improve the transparency (*white-boxness*) by providing domain context to the analysis as described in recommendations ***R6-annotation*** and ***R7-described***. Such annotations can range from adding textual descriptions and tags, to marking data with more systematic and well-defined domain-specific ontologies such as EDAM as mentioned above and BioSchemas [270].

If input data is not shared at *Level 1*, *primary authors* should ensure that recommendation ***R19-example*** is met, and provide example data. Such data can aid *secondary users* in analysing the methods shared and ultimately understanding their results. At *Level 3*, the information from previous levels combined with specific metadata related to data artefacts facilitates higher level classification of the workflow steps. This can be through motifs [271] such as data retrieval, pre-processing, analysis and visualisation. *Level 3* should ideally provide a researcher-centric view of data and enable *secondary users* to re-enact a set of related steps by providing a filtered and annotated view of the workflow enactment. Achieving *Level 3* is non-trivial for workflow definition and sharing, as it requires guided user annotations with controlled vocabularies. However, reusing related tooling and existing efforts such as BioCompute Objects [169] and DataCrate [272] can help to devise a strategy to satisfy the many provenance requirements of this level and hence to support comprehensive understanding of given analyses.

[8] http://bioportal.bioontology.org/ontologies/EDAM

Communicating resource requirements (***R19-resource-use***) at *Level 3* would involve domain-specific models for hardware use and cost prediction, as suggested for dynamic cloud costing [273] in *BioSimSpace* [274], or predicting assembler and memory settings through machine learning of variables like source biome, sequencing platform, file size, read count and base count in the *European Bioinformatics Institute (EBI) Metagenomics* pipeline [275]. For robustness such models typically need to be derived from resource usage across multiple workflow runs with varied inputs, e.g. by a multi-user workflow platform. Taking advantage of *Level 3* resource usage models might require pre-processing workflow inputs and calculations in an environment like R or Python, and so we recommend that models are provided with separate sidecar workflows for interoperable execution before the main workflow.

4.5 Conclusions

With any published scientific research, statements such as *"Methods and data are available upon request"* should not be acceptable in a modern open-science-driven research community. Considering the collaborative nature and emerging openness of bioinformatics research, it is essential to understand the significance of various artefacts in scientific workflow-centric studies. This impacts on their design, exchange and re-use. Given the huge heterogeneity in bioinformatics workflow design and implementation, the granularity and completeness of workflow provenance and associated resources also varies greatly. Each artefact can have a unique role in different use-cases depending on the audience, e.g. bioinformaticians, computer scientists, editors, reviewers or biologists. In this Chapter we identify key artefacts related to bioinformatics workflow-centric studies that impact on their provenance and hence on their trustworthiness, transparency, un-

derstandability and reproducibility.

Through the empirical study in *Chapter 3*, we identified a range of community best practices in Table 4.2 and elicited numerous associated artefacts in Section 4.3. Building on this, we presented a generalised framework defining hierarchical levels of provenance and associated resource sharing. Each level of this framework addresses specific recommendations and supports the capture of rich provenance information, with the topmost layer enabling the most complete sharing of provenance information for re-enacting workflows utilising domain-specific provenance. The range of granularity of provenance should tackle empty statements like "Methods are available" and differentiate them from "Methods are reproducible". By explicit enumeration of the levels of provenance support, it should be possible to quantify and directly assess the effort required to re-use a workflow and reproduce experiments directly.

Requiring researchers to achieve the levels defined in Section 4.4 individually is unrealistic without guidance and direct technical support. Ideally, the conceptual meaning of these levels would be translated into practical solutions utilising available technologies. In the next chapter, we show how it is possible to annotate resource aggregations and augment them with the retrospective provenance of workflow. We also describe the associated standards applied in this process.

CHAPTER 5

CWLProv –FORMAT OF WORKFLOW ENACTMENT REPRESENTATION

"An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures." –Jon Claerbout

5.1 Introduction

In *Chapter 4*, best practices articulated by the scientific community with respect to workflow-centric studies were pragmatically analysed to identify crucial artefacts that influence the provenance granularity, comprehensive exchange and sharing of bioinformatics workflow-centric studies. These artefacts are vitally important to address specific provenance applications and should be documented and captured as part of provenance, irrespective of the choice of workflow definition and implementation approach employed. With the scientific paradigm shifting towards open science, especially in the bioinformatics domain, information about the identified artefacts used in and supporting publications as a whole is encouraged by journals, fellow researchers and authors themselves. How-

ever, there still exists a gap in the consolidated understanding of various choices made during the workflow life-cycle such as the workflow definition approach, software licensing, data sources and versions of repositories used for sharing. All these choices have an impact on the granularity and completeness of the workflow provenance and should be considered when designing any workflow-centric study.

To understand the effects of these artefacts, we presented a novel hierarchical framework in *Chapter 4* defining levels of provenance and resource sharing of workflow-centric studies. Each level of this framework addresses both distinct and overlapping provenance demands depending on the captured provenance information and the aggregation level of the associated artefacts. Such levels can be used when sharing an analysis, whereby authors of a given study can convey valuable information about the state of the provenance of a given analysis.

Given the heterogeneity of workflow definition approaches, it is expected that the proposed framework, when translated into practical solutions, will also naturally result in varying workflow-centric solutions tied to specific WMSs. The heterogeneity could be of different forms that impact on the degree of provenance granularity, on the structure of the shared resources and the nature of the annotations. All of these changes creates another layer of complexity requiring considerable time and effort on the part of future audiences (researchers) to understand the varying nature of these workflow-centric commodities to actually benefit from published studies. The aim of this chapter[1] is to present a common format that can be used as a guideline to minimise such complexities.

[1]Elements of this chapter are submitted in the following article:
Khan, Farah Zaib and Soiland-Reyes, Stian and Sinnott, Richard O. and Lonie, Andrew and Goble, Carole and Crusoe, Michael R. **"Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv".** In: GigaScience (10.5281/zenodo.1966881)

To support interoperability of the provenance levels defined in Section 4.4, we present ***"CWLProv"***, a format for machine-readable provenance representation and structured resource aggregation leveraging open standards. These standards are selected due to their technology-independent nature resulting in interoperable solutions that abstract from the lower level workflow definition details and associated provenance documentation as well as the aggregated resource annotation capabilities that these standards offer. We follow the recommendation *"Reuse vocabularies, preferably standardised ones"* [276] from best practices associated with data sharing, representation and publication on the web to achieve consensus and interoperability of workflow-based analyses. We demonstrate the applicability of *CWLProv* by extending an existing workflow executor. This chapter includes details of the design principles of the applied standards that form the core structure of *CWLProv*. We also present the details of the *CWLProv* realisation.

5.2 Applied Standards and Vocabularies

Considering the impact and significance of open source licensing discussed in Section 3.4.6 and 4.3.9, we have used open licensing and community-driven initiatives as metrics when choosing the required standards and technologies for *CWLProv*. In order to avoid a format that is tied to a specific workflow platform, *CWLProv* incorporates best practices and standards that support a more abstract representation of analyses. In addition, we integrate domain-agnostic but extensible standards such that the solutions are not specialised for a single scientific domain, e.g. bioinformatics, but can be extended to fit the needs of any discipline.

CWLProv integrates three core standards as the basis for devising a common format for representing workflow enactment and its retrospective provenance.

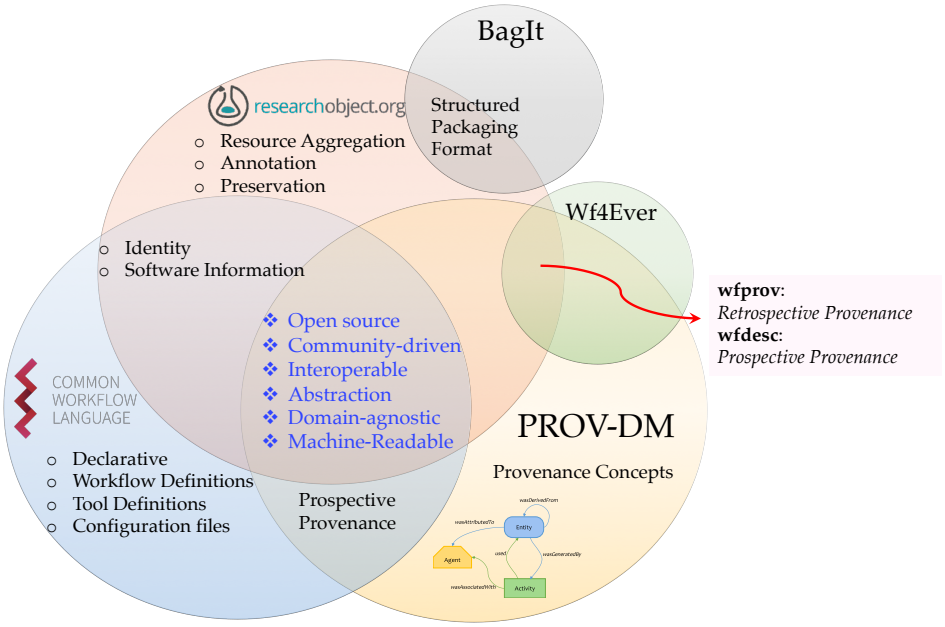

Figure 5.1: Standards integrated in *CWLProv* including their individual functionalities and their overlapping principles.

Specifically, it leverages the Common Workflow Language (CWL) for workflow definition, Research Objects (ROs) for resource aggregation and the PROV-Data Model (PROV-DM) to support retrospective provenance associated with workflow enactment. In addition, *BagIt* is used for serialisation of ROs and few *Wf4Ever* ontologies are also utilised. The distinctive and overlapping functions along with the properties and principles of these standards are depicted in Figure 5.1. This section includes the details of these standards and their functionalities that have directly shaped *CWLProv*.

5.2.1 Common Workflow Language (CWL)

CWL is a community-driven standard for declarative scientific workflow descriptions. It has been developed by a multi-vendor working group comprised of vari-

ous organisations and individual contributors. With roots in "make"[2], CWL tasks are executed based on their dependencies making the workflow a conventional scientific *data-flow* process. Because of the isolation and explicit nature of each task, it is possible to design large-scale workflows that are scalable and portable across different computing environments ranging from personal workstations to cluster, Cloud and High Performance Computing (HPC) environments. CWL has been widely adopted by many workflow design and execution platforms to support interoperability across diverse platforms. Examples include Toil, Arvados, Rabix [277], Cromwell [194], REANA[3], and Bcbio [182] with implementations for Galaxy, Apache Taverna and AWE[4] currently in progress. CWL is not tied to a specific operating system, technology or WMS which makes it an ideal vehicle to fulfil ***R17-format*** (Table 4.1) and overcome the heterogeneity issues of workflow platforms.

CWL is defined with a schema, specification and test suite that provides declarative constructs. It adopts a pragmatic approach to define workflow and command line tool descriptions. As demonstrated in *Chapter 3*, it makes minimal assumptions about base software dependencies, configuration settings, software versions, parameter settings and execution environment. It's underlying object model supports the explicit declaration, comprehensive recording and capture of information about each task and the involved artefacts. This information can subsequently be extracted, structured and documented as part of prospective as well as retrospective provenance during a given workflow's life-cycle. This can be published alongside any resultant analysis using that workflow. In the following section, the structure of a typical CWL workflow and command line tool, referred to as a "CWL process", is described.

[2]http://man7.org/linux/man-pages/man1/make.1.html
[3]http://www.reana.io/
[4]https://github.com/MG-RAST/AWE

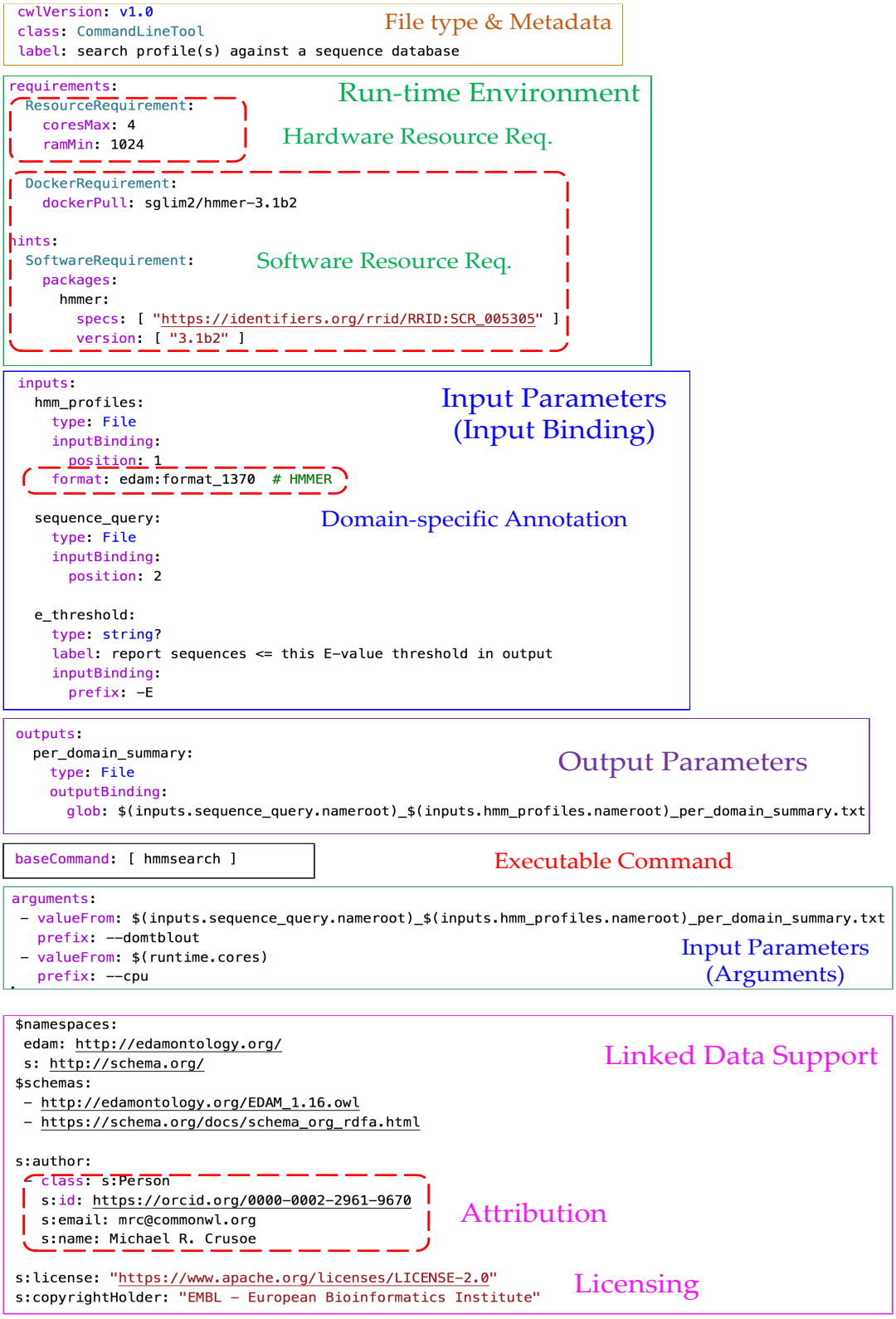

Figure 5.2: Components of a command line tool description for *hmmsearch* [279].

5.2.1.1 CWL Process Description Document

Figure 5.2 shows an example CWL command line tool description file for *hmmsearch*, a tool to search a sequence database for sequence homologs using probabilistic models, e.g. profile Hidden Markov Models (profile HMMs) [278]. Each section of the tool description is highlighted using different colours and tagged to differentiate between the various constructs.

5.2.1.2 Declarative Constructs

As a standard, CWL provides different constructs to handle the artefacts identified in Section 4.3 that are used to support provenance capture. This includes the declaration of the software and hardware resources used in any workflow or command line tool specification which has to be satisfied (resolved) by the WMS supporting the enactment. The responsibilities of the WMS supporting the CWL workflow enactment include scheduling the jobs depending on the availability of resources, setting up the run-time environment based on the declared requirements, checking the availability of inputs as specified in the input object and collecting the final data outputs. The following are the specific structures and approaches provided by CWL to support the declaration of different artefacts of a workflow that have been utilised by *CWLProv* for provenance capture.

5.2.1.2.1 Input Parameters

The input parameters of a CWL workflow can be provided in a combination of ways including:

- Default values as part of the CWL Process description file;

- Provided through the command line at run-time, or by
- Supplying a YAML/JSON input object document which can be optionally marked as executable by including a SciCrunch[5] Internationalised Resource Identifier (IRI) that references the CWL Process document to be enacted.

The WMS or workflow executor supporting CWL workflow enactments can aggregate different parameters if supplied in more than one way to make one input object that can be published with a given analysis. An example is *cwltool* [234], a reference implementation for enacting CWL workflows which gathers parameters provided using a combination of the above approaches into one input object. As discussed in Section 4.4.1, *Level 0* requires provision of the configuration settings of a workflow, hence this feature can be used to obtain and publish user-defined parameter settings that may be used in a given analysis.

5.2.1.2.2 Software Environment Declaration

CWL descriptions can contain information about software requirements including the underlying tools used such as Docker, Singularity, Conda, Debian or any other packaging system. There are two ways to explicitly declare such information:

- ***SoftwareRequirement***: this provides a list of software packages required to be configured in order to enact a given CWL process. The software requirement can either be declared as part of "hints" or as strict "requirements" resulting in a failed enactment if the requirement is not satisfied by the WMS. CWL encourages the provision of an IRI for the software packages which can be used to invoke a configuration action using a given package/configuration manager.

[5]https://scicrunch.org/

- ***DockerRequirement***: this requirement contains information about how to fetch/build the image of the Docker container used to enact a given CWL process.

Both of these constructs (as shown in Figure 5.2) are intended to be utilised by workflow authors to explicitly define the execution environment by declaring the required software packages and their versions needed for any CWL process. The containers specified using *DockerRequirement* should be designed by following recommendations for containerising proposed in [156]. These requirements are expected to be managed by the WMS enacting the workflow.

5.2.1.2.3 Hardware Resource Requirement

As shown in Figure 5.2, workflow authors can also specify hardware resource requirements as "requirements" or "hints" where the minimum amount of resource "MUST" be available to enact a given CWL process such as:

- Minimum / maximum number of required cores;
- Minimum / maximum required RAM;
- Minimum / maximum reserved storage requirements for temporary intermediate outputs, and the
- Minimum / maximum reserved storage for the workflow output.

These resources when declared as part of workflow specification provide an a priori estimation of the computational resources needed for enacting a given workflow.

```
#!/usr/bin/env cwl-runner
doc: |
    TOPMed RNA-seq CWL workflow.
    Documentation on the workflow can be found
    [here](https://github.com/heliumdatacommons/cwl_workflows/blob/master/topmed-workflows/TOPMed_RNAseq_pi
    Example input files:
    [Dockstore.json]
    (https://github.com/heliumdatacommons/cwl_workflows/blob/master/topmed-workflows/TOPMed_RNAseq_pipeline/
    [rnaseq_pipeline_fastq-example.yml]
    (https://github.com/heliumdatacommons/cwl_workflows/blob/master/topmed-workflows/TOPMed_RNAseq_pipeline/

    Quickstart instructions are [here]
    (https://github.com/heliumdatacommons/cwl_workflows/blob/master/topmed-workflows/TOPMed_RNAseq_pipeline/

    [GitHub Repo](https://github.com/heliumdatacommons/cwl_workflows)

    Pipeline steps:
    1. Align RNA-seq reads with [STAR v2.5.3a](https://github.com/alexdobin/STAR).
    2. Run [Picard](https://github.com/broadinstitute/picard) [MarkDuplicates](https://broadinstitute.github
    2a. Create BAM index for MarkDuplicates BAM with [Samtools 1.6](https://github.com/samtools/samtools/rel
    3. Transcript quantification with [RSEM 1.3.0](https://deweylab.github.io/RSEM/)
    4. Gene quantification and quality control with [RNA-SeQC 1.1.9](https://github.com/francois-a/rnaseqc)

cwlVersion: v1.0
class: Workflow
label: "TOPMed_RNA-seq"
```

Figure 5.3: Example description included as part of a workflow [280].

5.2.1.2.4 Metadata & Domain-specific Annotations

A CWL process document can contain user-defined descriptions as well as established ontologies to describe the background and intention of the workflow and the expected file formats. User-defined textual descriptions can be added as short human-readable *"labels"* or longer descriptions as *"docs"* and associated with any CWL process object including the document itself, the inputs and outputs. A snippet of a detailed description of a workflow functionality for RNA-seq data analysis is shown in Figure 5.3.

Workflow authors can extend input and output parameters to include domain relevant information related to the input/output formats to provide specific context. An example is shown Figure 4.1 and 5.2, where the bioinformatics specific EDAM ontology is used to specify the accepted/required formats of the input

parameters.

5.2.1.2.5 Attribution & Licensing

CWL provides linked data support to include attribution details of workflow authors, contributors and maintainers leveraging existing ontologies such as schema.org[6]. These details can include an unambiguous identifier such as an ORCID or an email address etc. In addition, workflow authors can include associated citation details such as the date of creation, license details and the location of the CWL process document. A permissive licence such as Apache 2.0 is recommended as it supports reuse of tools and workflows to prevent the community from repeating development effort. An example of author details and licensing is shown in Figure 5.2.

5.2.1.2.6 Graphical Representation

A CWL viewer [281] is available as a *de facto* standard web visualisation used to display data-flow complex workflow graphs that would otherwise be challenging to comprehend. A colour-coded DAG (Figure 2.3) differentiating between different components of the workflow available on GitHub, can be created and downloaded in different formats. In addition, the descriptions extracted from the CWL workflow document such as *label* and *doc* strings, IDs and format types for inputs/outputs and steps can be displayed in a table that provides a good representation instead of requiring manual parsing of the CWL document. The workflow DAG generated by the CWL viewer, although contributing to the completeness of the prospective provenance, differs from what was discussed in Section 4.3.4 as it is not an abstract representation but rather a translation of a given workflow document to a graphical representation.

[6]https://schema.org/

All of these constructs aid in the capture of key artefacts identified in *Chapter 4* and can be used as part of prospective provenance. Recommended practices to workflow design[7] also emphasise the inclusion of these details in a given CWL process document, be it given as a workflow or command line tool.

5.2.2 Research Object (RO)

A Research Object (RO) encapsulates the digital artefacts and any/all semantic annotations associated with a given data-driven computational analysis into one unit. This enables its subsequent sharing and preservation. The aggregated resources of any workflow-centric study can include: input and output data for validation of analysis results; computational methods such as command line tools and workflow specifications; attribution details regarding users; retrospective as well as prospective provenance, and machine-readable annotations related to the artefacts and relationships between them as shown in Figure 5.4.

The three core principles of the Research Object approach are to support **"Identity"**, **"Aggregation"**, and **"Annotation"** of research artefacts [282]. This supports the accessibility of tightly-coupled, interrelated and well-understood aggregated resources involved in a computational analysis and their use / re-use as identifiable objects through use of unique (persistent) identifiers such as DOIs and/or ORCIDs. The goal of the Research Object approach is to make any published scientific investigation and its output artefacts *interoperable, reusable, citable, shareable and portable*, hence it is well aligned with the idea of interoperable and platform-independent, provenance-enabled workflows. In the following sections, the Research Object ontologies utilised in the design of *CWLProv* and their serialisation format are described.

[7] https://www.commonwl.org/user_guide/recpractices/

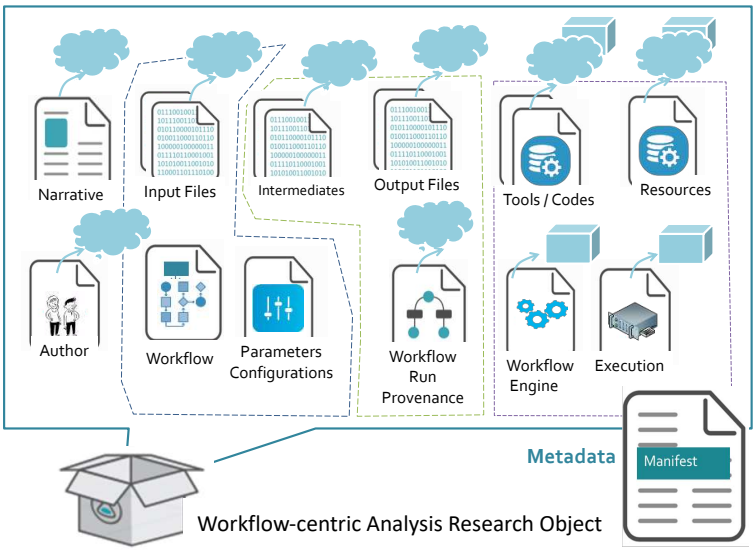

Figure 5.4: Resources typically aggregated in a workflow-centric Research Object (from [35]).

5.2.2.1 Specialised Workflow Provenance Ontologies

Workflow-centric ontologies are used to add metadata describing aggregated artefacts and their relationships [29]. Two such ontologies *"wfdesc"* and *"wfprov"* (described in Section 2.2.4.3) relate to retrospective and prospective provenance representation have been incorporated into *CWLProv*. In addition, The Research Object Ontology *"ro"* is also utilised for some cases. These ontologies are used in addition to the PROV-DM to represent the provenance of workflow-specific objects (discussed later in Section 5.4).

5.2.2.2 Serialisation Format –*BagIt*

While ROs can be serialised in several different ways, *CWLProv* reuses the BD-Bag approach based on *BagIt* [165], which has been shown to support large-scale workflow data storage and transfer demands [283]. This approach is also com-

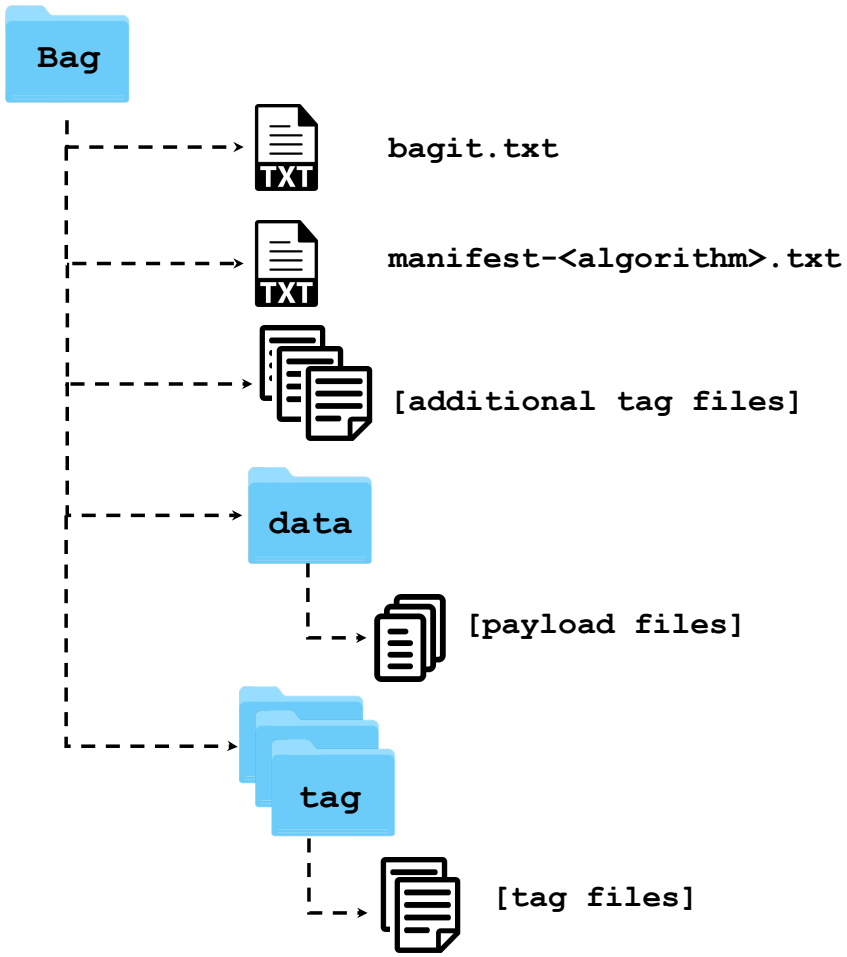

Figure 5.5: Example of a Valid BagIt structure adapted from [165].

patible with data archiving efforts from the Library of Congress, DataOne [165] NIH Data Commons[8], and the Research Data Alliance[9] (RDA).

An example of the hierarchical file layout of a typical *Bag* is shown in Figure 5.5. A valid Bag must contain a *"payload"* directory named as *"data/"*, a payload manifest named as *"manifest-algorithm.txt"* and a bag declaration file named as *"bagit.txt"*. In addition to these mandatory files, the bag can optionally include *"tags"* metadata files and tag directories where the term *tag* can be interpreted as optional content of the Bag. A minimal Bag must have a bagit.txt file, i.e. it

[8]https://github.com/NCATS-Tangerine/smartBag
[9]https://github.com/RDAResearchDataRepositoryInteropWG/kitdm-bagit-tool

may have an empty *data/* and thus an empty manifest file. We have specialised the *CWLProv* RO by including additional tag directories and files (discussed in Section 5.3).

5.2.3 PROV Data Model (PROV-DM)

The World Wide Web Consortium (W3C) developed PROV based on the recommendations of the Provenance Incubator Group [25]. PROV refers to a suite of specifications introduced to support the unified and interoperable representation and publication of provenance information on the Web. As discussed in Section 2.2.4.2, the underlying conceptual PROV Data Model (PROV-DM) provides a domain-agnostic approach designed to capture the fundamental features of provenance with support for various extensions used to integrate domain-specific information as shown in Figure 5.6.

The figure shows the relationship between various standards utilised to re-

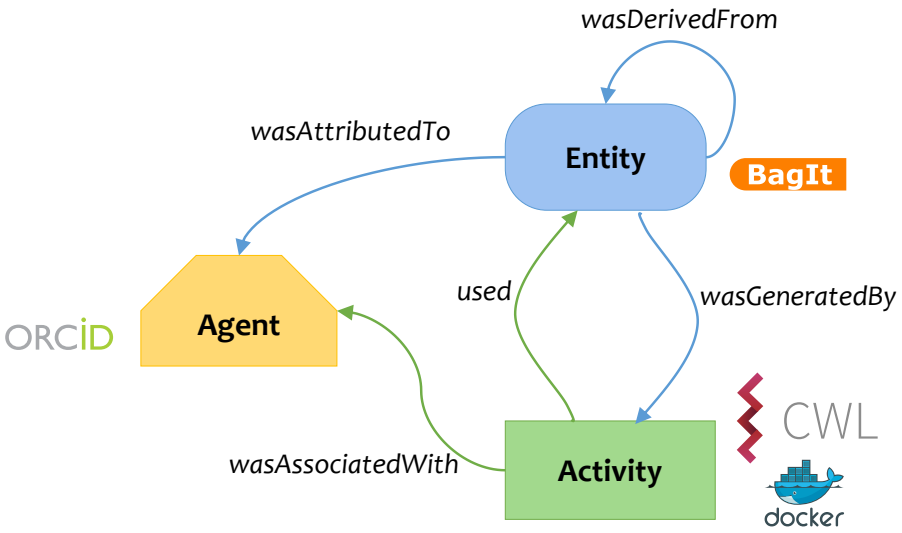

Figure 5.6: Core concepts of the PROV Data Model. Adapted from W3C PROV Model Primer [114].

alise the three core elements of PROV-DM including ORCIDs for researchers designing/enacting workflows; BagIt for stratifying the entities, and CWL to describe and standardise the workflow activities. We utilise two provenance serialisations of PROV in *CWLProv*: PROV-Notation (PROV-N) [284] and PROV-JSON [285]. PROV-N is designed to achieve serialisation of PROV-DM instances by formally representing the information using simplified technology-independent syntax to improve readability. PROV-JSON is a lightweight interoperable representation of PROV assertions using JavaScript constructs and data types that can be used for querying the information. The key design and implementation principles of these two serialisations of PROV are that they are understandable and interoperable, and hence preferred for the design of an adaptable provenance profile. The details of the individual constructs and their relationships in the context of *CWLProv* are discussed in Section 5.4.

5.3 *CWLProv* Format

A *CWLProv* RO complies with the BagIt format with specialised tag directories and files used to encompass the artefacts and annotations identified in Section 4.3 using targeted tools (Figure 5.7). It includes a *Workflow provenance profile* to capture the detailed retrospective provenance of CWL workflow enactment. For compliance with the BagIt file hierarchical structure, any BagIt tool or library can be used to verify the content and completeness of a given *CWLProv* RO. This section covers the details of the *CWLProv* RO specification defined for structuring any CWL workflow enactment. It is noted that this format can be applied using any workflow definition approach, provided that the approach supports explicit declarations of required artefacts as exemplified with CWL.

The *CWLProv* RO base directory must contain the following annotation files:

- bagit.txt which contains two lines recording the BagIt version number and a tag file for character encoding;
- bag-info.txt with metadata about the contents of the file including the date of creation, the WMS used and the format used to stratify the contents;
- manifest-sha1.txt file which provides the complete listing of the payload file names and their corresponding checksums. The payload file naming convention used in *CWLProv* format is discussed in the following Section 5.3.1.

In addition, the base directory can optionally contain tag manifest files in *tagmanifest-algorithm.txt* including a listing of other tag files and their checksums generated using the algorithm that the tagmanifest name contains. All of the files outside *data/* should be listed in the tag manifest according to the BagIt specification. The algorithm used in at least one of the tagmanifest files (if there are multiple files with different hashing algorithm) should be the same as the mandatory payload manifest file. The base directory must contain four sub-directories: *"data/"*; *"workflow/"*; *"snapshot/"* and *"metadata/"*. The following subsections describe the structure and contents of these sub-directories.

5.3.1 data/

data/ is the payload (collection) of all input datasets used in a given workflow enactment along with the resultant output files. Data must be labelled and identified by a checksum value based on a checksum algorithm registered in IANA's "Named Information Hash Algorithm Registry" [286]. The use of content addressable storage [287] simplifies identifier generation for data. It also permits data integrity checking and helps to avoid local dependencies, e.g. hard-coded

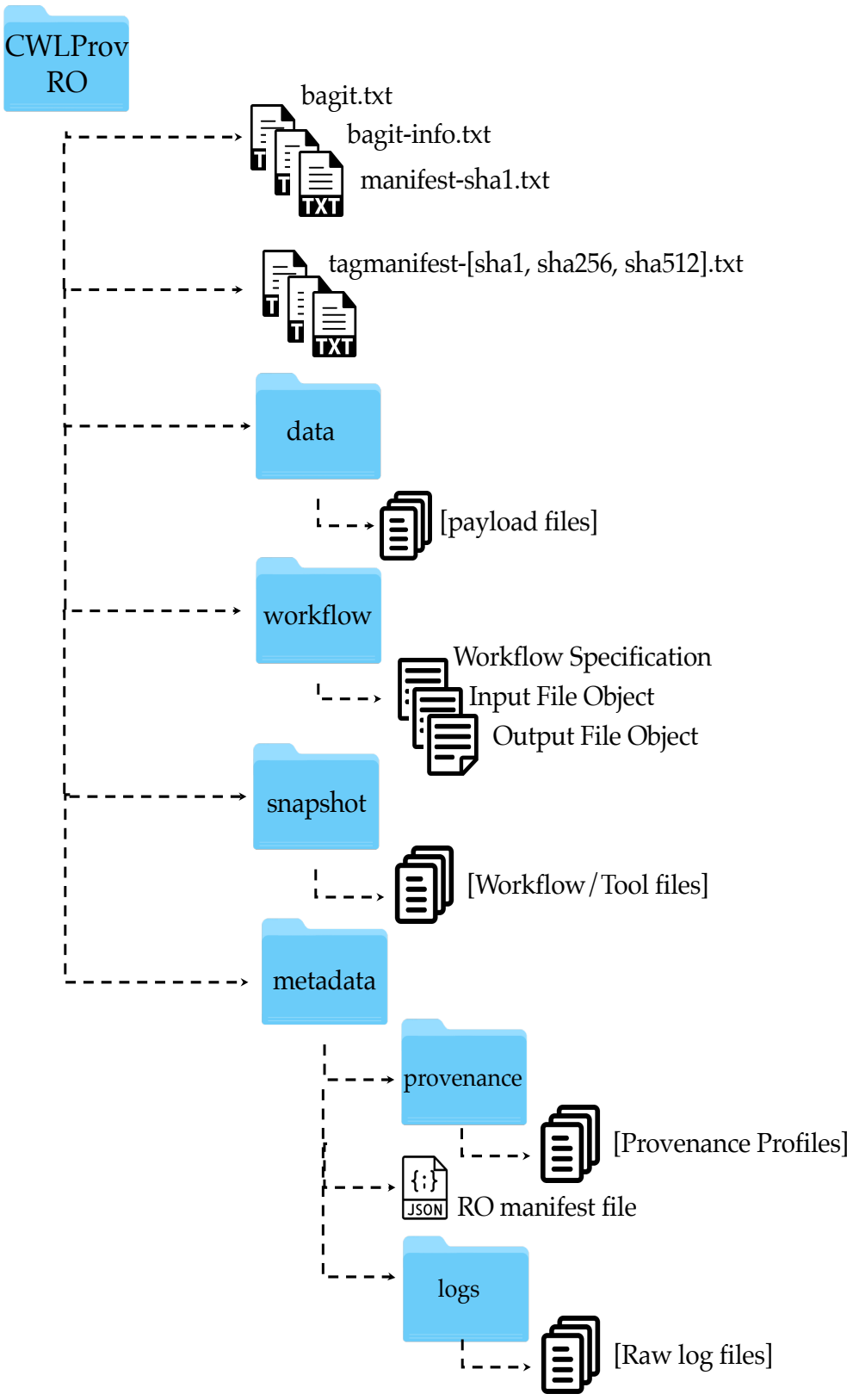

Figure 5.7: Customised BagIt structure with tagged directories and files given as *CWLProv* ROs.

file names. This practice minimises the dependence on local file system, user-defined file names and localised paths that need to be resolved when a Research Object is shared.

Workflow platforms might use customised identifiers for file objects. In this case, it is advised to use such identifiers as file names to store data in order to avoid redundancy and to comply with the system/platform used to enact the workflow. Hard-coded file paths should still be avoided. The checksums of the payload files in the *data/* directory must be listed in the BagIt manifest-*.txt (where the asterisk refers to the hashing algorithms) as well as in the RO manifest as JSON-LD [288] that is stored in *metadata/manifest.json* (described in Section 5.3.4.

5.3.2 workflow/

CWLProv ROs must include a WMS-independent executable version of the workflow in the *workflow/* directory. When using CWL, this directory should contain the complete workflow specification file, one or more input file objects with parameter settings used to enact the workflow and an output file object. The output is used for understandability and not for re-enactment. Actual output files may not exactly match the files that were enacted, e.g. absolute paths in the input job file are recommended to be replaced with relativised content-addressed paths referring to the input and output file objects in the *data/* directory.

The input file object should capture all dependencies of the input data files including any index files associated with BAM files and reference genome sequence files. It is possible to reference a directory in the input file object instead of individual input data files. In that case, the directory reference should be expanded

to list the actual files to explicitly declare and capture the contents utilised in a given workflow enactment. The output file object included in *workflow* should contain details of the workflow outputs. In the case of a CWL workflow using *cwltool*, this is a JSON file with a list of outputs and their associated metadata such as their location, their checksum and the size of the files.

cwltool supports aggregation of CWL workflows and any referenced external descriptions (such as sub-workflows or command line tool descriptions) into a single executable file. This feature is leveraged in the *CWLProv* implementation (details in Section 5.5) to rewrite the workflow files making them re-executable without depending on workflow or command line descriptions outside the associated RO. Other workflow design and implementation approaches should adopt similar features to create executable workflow objects.

5.3.3 snapshot/

snapshot/ comprises copies of the workflow and command line tool specifications files "as-is". It is recommended to use these resources only for understanding the workflow enactment, since these files might contain absolute paths or be host-specific, i.e. they may not necessarily be re-enactable elsewhere. However, these specifications may be "re-used" in a similar analysis in a similar situational context.

5.3.4 metadata/

This sub-directory should contain RO manifest files as JSON-LD using two sub-directories *metadata/logs* and *metadata/provenance*. In the following sub-sections,

details about the structure of each element are included.

5.3.4.1 RO Manifest

The RO manifest file should record a range of metadata including the standard[10] that it conforms to, the WMS that created the RO, the researcher responsible for the workflow enactment and the date of creation. The *"aggregates"* section of the file must list all resources aggregated by the RO. The elements of a given aggregate include the date of creation, a unique Uniform Resource Identifier (URI), the name, the media type, any conforming standard and the folder. *aggregates* can contain links to both embedded as well as external resources. The *"annotations"* section should include information related to the Research Object or any of the aggregated resources. *manifest.json* follows the structure defined for RO Bundles [289] however *.ro/* is referred to as *metadata*. Further detail about the manifest file contents is documented on GitHub as part of the *CWLProv* specification [290].

5.3.4.2 logs/

Any raw enactment log information should be made available in *metadata/logs*. This should include the actual commands executed and parameters used for each step along with time-stamps for each operation. In the case of distributed computing environments, this directory can possibly contain multiple log files labelled with unique identifiers referring to the execution engines hosted potentially at geographically distributed resources.

[10] e.g. https://w3id.org/cwl/prov/0.6.0

5.3.4.3 provenance/

Retrospective provenance profiles for the workflow execution associated with Research Objects require rich metadata. These profiles should exist in *metadata/provenance*. It is recommended to make the availability of a *primary* profile mandatory and this should conform with the PROV-N format. This file describes the top-level workflow execution. As described in *Level 2* (Section 4.4.3), it is quite possible to have nested workflows. In this case, a provenance profile for all such workflows should be included in this directory. If there are additional formats of the provenance files such as JSON, JSONLD, XML etc, these should be included in the said directory and a (compulsory) declaration *"conformsTo"* should be provided to document their formats and dependencies in the RO manifest. A nested workflow profile should be named such that there is a link between the respective step in the primary workflow and the nested workflow ideally using unique identifiers.

As the PROV-DM has a generalised structure, there might be some provenance aspects that are specific to particular workflows that are hard to capture. In this case, ontologies such as *wfdesc* [291] can be used to describe the abstract representation of the workflow. Use of *wfprov* [292] to capture workflow provenance aspects is also encouraged. These two ontologies are used in the provenance profile described in Section 5.4. Alternative extensions such as ProvOne [293] can also be utilised if the WMS or workflow executor is using these extensions already.

5.3.4.4 URI scheme

CWLProv reuses Linked Data standards like JSON-LD, W3C PROV and Research Objects. A challenge with Linked Data in distributed and desktop computing is how to make identifiers that are absolute URIs and hence globally unique. For example, for *CWLProv* a workflow may be executed by an engine that does not know where its workflow provenance will be stored, published or finally integrated. To address this *CWLProv* generators should use the proposed arcp [294] URI scheme to map local file paths within the RO BagIt folder structure to absolute URIs for use within the RO manifest and hence with associated PROV traces. A base URI is included in the RO manifest *context* that serves as the root of the RO. This URI should be based on the identifier of the workflow enactment. This can be used to resolve relative URI references.

Consumers of *CWLProv* ROs that do not contain an arcp-based External Identifier can generate a temporary arcp base to safely resolve any relative URI references not present in the *CWLProv* RO folder. Implementations processing a *CWLProv* RO may convert arcp URIs to local *file:///* or *http://* URIs depending on how and where the *CWLProv* RO was saved, e.g. using the "arcp.py" library [294]. The relationship between the different components of a CWL workflow enactment with the structure of a *CWLProv* RO is shown in Figure 5.8.

5.4 Retrospective Provenance Profile

Retrospective provenance profiles supports data-flow workflow enactment using provenance models with at least one needing to support PROV-N serialisation. The provenance standards utilised in the design of this profile are highly extensible making it possible to incorporate other standards if required by different

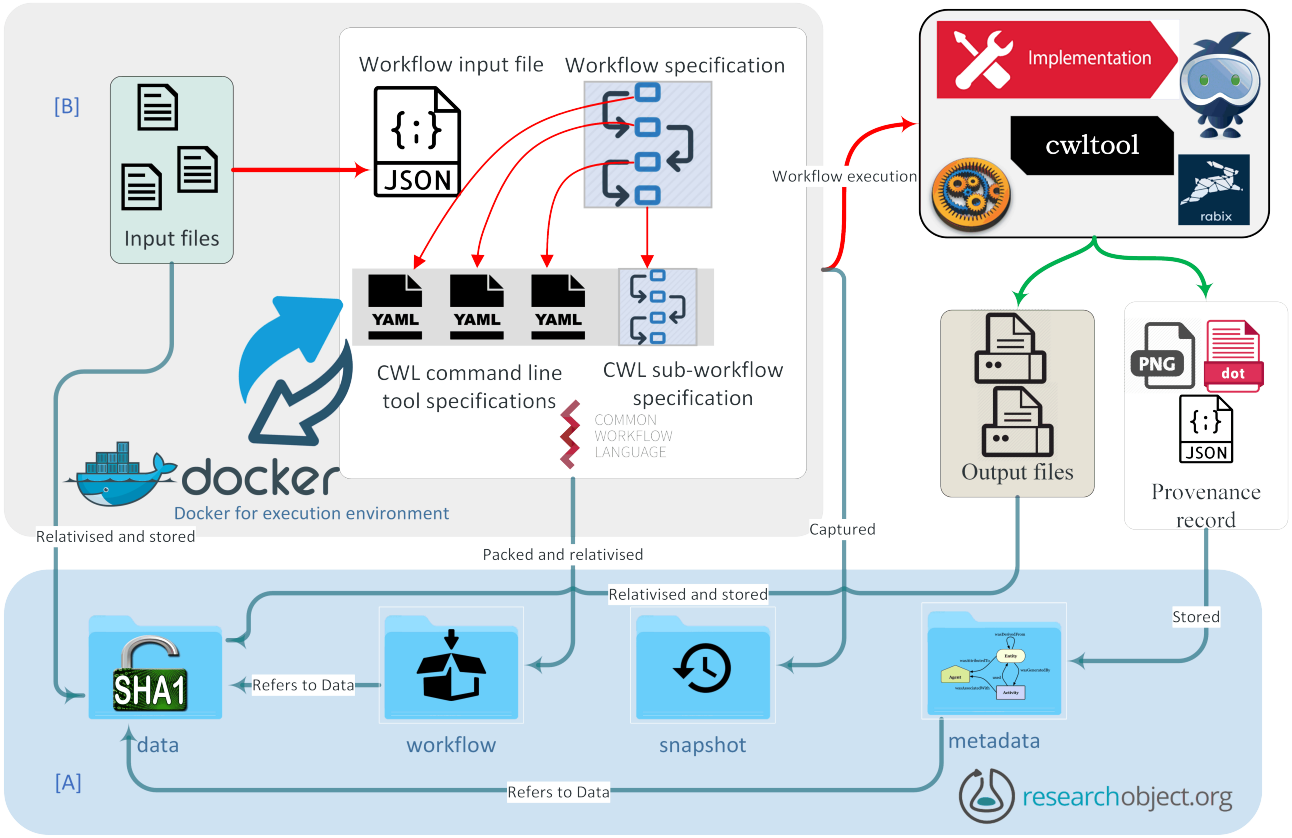

Figure 5.8: High-level schematic representation of the aggregation and links between the components of a given workflow enactment. The layers of execution are separated for clarity. Workflow specification and command line tool specifications are described using CWL. Each individual command line tool specification can optionally interact with Docker to satisfy software dependencies. [A] The RO layer shows the structure of the Research Object including its content and interactions with different components in the Research Object and [B] the CWL layer.

WMS or scientific domains. In this section, we introduce some key features used in the structure of the retrospective provenance profile for a given CWL workflow enactment using *CWLProv*. The features are not tied to any platform or workflow definition approach and hence can be used to document retrospective provenance of any workflow. The components of the profile with respect to CWL workflow enactment are summarised in Table 5.1 and detailed documentation is available on GitHub[11].

5.4.1 wfdesc

As PROV is a general standard, it lacks features to relate a plan (i.e. a workflow description) with sub-plans. To tackle this, we use *wfdesc:Workflow* (a subclass of *Prov-Plan* for the workflow specification and *wfdesc:hasSubProcess* to relate individual steps of the workflow. Each step of a given workflow is described by *wfdesc:Process* where a Process refers to a workflow step in the RO ontology. This provides the prospective view of the workflow specification without the requirement for actually executing it.

5.4.2 wfprov

This term is used for the activities and entities used in a given workflow enactment that relate to retrospective provenance. *"wfprov:WorkflowRun"* and *"wfprov:ProcessRun"* represent the workflow and individual command line tool execution components respectively. Data items are described by *"wfprov:Artifact"*, which is a subclass of *prov:Entity*.

[11] https://github.com/common-workflow-language/cwlprov/blob/master/prov.md

5.4.3 Entity

An Entity is used to refer to any data artefact, input configuration file, workflow or command line tool specification. Each data artefact is identified using hash values depending on the hash algorithm used to store files in the *"data/"* directory. The provenance profile predominantly stores retrospective provenance components including entities used for the workflow specification and labelled steps used for understandability. These entities are utilised in *wfdesc:Process* and *wfdesc:hasSubProcess* and are labelled as *prov:label="Prospective provenance"*. For an input parameter with an absolute value such as a string, integer or a list, the entity contains the *prov:value* as an attribute used to store the actual value. For inputs of file type, the entity is labelled with *wfprov:Artifact*.

It is possible that a workflow or workflow step requires additional files, e.g. index files associated with the human reference sequence. These files are referred to as "SecondaryFiles" in the CWL specification. In the *CWLProv* profile, these files are declared through *"cwlprov:SecondaryFile"* and the derivation relationship *wasDerivedFrom* between the secondary and primary file is used. Moreover, in case of directory type inputs representing a collection of named files or subdirectories, the concept of *prov:Dictionary*[12] is utilised to explicitly declare the contents of the directory in the provenance profile and hence to refer to the individual objects using checksum-based identifiers. If a directory input is represented in this way, a collection membership of directory membership, e.g. group, should also be provided using *hasMember*. These two approaches are adopted for secondary files and directory structure inputs and subsequently help to record fine-grained provenance information about the data artefacts involved in a given workflow enactment.

[12]https://www.w3.org/TR/prov-dictionary/

5.4.4 Activity

Workflow enactment or command line tool invocations are classified as activities that can be identified by a Universal Unique Identifier (UUID). They can also be labelled with the absolute name of the step given in the workflow file to improve the readability. These activities are labelled using the above described *wfprov:WorkflowRun* and *wfprov:ProcessRun* to differentiate between workflow and command line tool-based enactments.

5.4.5 Agent

An agent in *CWLProv* provenance profile is either an *"Agent"* or a *"SoftwareAgent"* [295]. A *SoftwareAgent* can refer to the engine used to enact the Workflow Plan and should be identified by a unique identifier, e.g. a UUID. It is labelled using *wfprov:WorkflowEngine*. If a Docker container is utilised in any step, it should be documented in the profile as a *SoftwareAgent*. The user enacting the workflow is referred to as an *Agent* [295] and used to store attribution details, e.g. through ORCID identifiers.

5.4.6 wasAssociatedWith

An association is represented as *wasAssociatedWith* in PROV-N. It is used to assign a responsibility to an agent for a given activity. It relates activities such as *WorkflowRun* and *ProcessRun* to the *SoftwareAgent*.

Table 5.1: *CWLProv* profile of W3C PROV, extended with Research Object Model's *wfdesc* and *wfprov*.

PROV type	Subtype	Relation	Range
Plan	wfdesc:Workflow wfdesc:Process	wfdesc:hasSubProcess	wfdesc:Process
Activity	wfprov:WorkflowRun	wasAssociatedWith ↳ hadPlan	wfprov:WorkflowEngine wfdesc:Workflow
		wasStartedBy ↳ atTime	wfprov:WorkflowEngine *ISO8601 timestamp*
		wasStartedBy	wfprov:WorkflowRun
		wasEndedBy ↳ atTime	wfprov:WorkflowEngine *ISO8601 timestamp*
	wfprov:ProcessRun	wasStartedBy ↳ atTime	wfprov:WorkflowRun *ISO8601 timestamp*
		used ↳ role	wfprov:Artifact wfdesc:InputParameter
		wasAssociatedWith ↳ hadPlan	wfprov:WorkflowRun wfdesc:Process
		wasEndedBy ↳ atTime	wfprov:WorkflowRun *ISO8601 timestamp*
	SoftwareAgent	wasAssociatedWith ↳ *CWLProv*:image	wfprov:ProcessRun *docker image id*
SoftwareAgent	wfprov:WorkFlowEngine	wasStartedBy	Person *ORCID*
		label	*cwltool −−version*
Entity	wfprov:Artefact	wasGeneratedBy ↳ role	wfprov:Processrun wfdesc:OutputParameter
Collection	wfprov:Artefact	hadMember	wfprov:Artefact
	Dictionary	hadDictionaryMember ↳ pairKey	wfprov:Artefact *filename*

Indentation with ↳ indicates n-ary relationships which are expressed differently depending on PROV syntax.

5.4.7 wasStartedBy

The start of an activity is represented as *wasStartedBy* in PROV-N. This is used to denote when an activity was either started by an entity (trigger) or another activity (starter). It records an activitys identifier, the starter and the time of commencement. In the case of a *WorkflowRun*, the starter is not specified as the workflow enactment is not triggered by another activity, whilst in the case of a *ProcessRun*, the corresponding *WorkflowRun* is specified as the starter.

5.4.8 Used

In PROV-N, *used* is used to represent the "Usage", i.e. when an entity participates in an activity. *WorkflowRun* and *ProcessRun* both utilise this concept so that the same entity representing a data artefact can be used by *WorkflowRun* and *ProcessRun* albeit at different instances of time. To improve readability, an optional parameter *prov:role* is added to *used* to assign the name (string) given by users to certain inputs and outputs in a workflow description.

5.4.9 wasGeneratedBy

The concept of generation is represented as *wasGeneratedBy* in PROV-N. This refers to the situation where an entity is created (generated) by an activity and can only be used after its generation. It associates all output data artefacts, represented as entities with the respective activity that generated them. It is noted that the same entity can be generated by a *ProcessRun* as well as the parent *WorkflowRun* if it is declared as an output at both levels.

5.4.10 wasEndedBy

wasEndedBy is used for the concept "end", i.e. the time when an activity was ended by another activity (ender) or by an entity (trigger). In the provenance profile, it represents the relationship between *ProcessRun* and *WorkflowRun* and the time when the *ProcessRun* was ended by the *WorkflowRun*. In addition, the relationship between *WorkflowRun* and *SoftwareEngine* is also documented using this property since each workflow enactment is triggered and terminated by the engine itself.

5.4.11 has_provenance

As described in Section 4.4.3, in order to support *white-box* provenance, the inner workings of nested workflows should be included in provenance traces. If a step represents a nested workflow, a separate provenance profile should be included in the RO. Moreover, in the parent workflow trace, this relationship is recorded using *has_provenance* as an attribute of the *Activity* step, i.e. this refers to the respective profile of the nested workflow.

A provenance profile can be modified and adapted to include constructs from other specialised standards to cater for domain-specific requirements, e.g. *PROV-Wf* for HPC environments and D-PROV for prospective provenance of data-flow models.

5.5 Practical Realisation of *CWLProv*

CWLProv [290] provides a format for representing any given workflow enactment and its provenance that can be adopted by any workflow executor or platform, provided that the underlying workflow definition approach is at least as declarative as CWL, i.e. it captures the necessary components described in Section 5.2.1.2. In the case of CWL, as long as the conceptual constructs are common amongst the available implementations and executors, a workflow enactment can be represented in *CWLProv* format. The practical demonstration of the proposed format helps in understanding the structure and exploring the properties and functions offered by *CWLProv* format. To that end, this section describes the practical realisation using a Python-based reference implementation of CWL *cwltool*, also used in *Chapter 3* for enacting the variant calling workflow.

5.5.1 Reference Implementation

cwltool is a feature complete implementation of CWL. It provides extensive validation of CWL files as well as offering a comprehensive set of test cases to validate new modules introduced as extensions to the existing implementation. Thus it provides the ideal choice for implementing *CWLProv* and demonstrating provenance support and resource aggregation. A range of existing classes and methods were utilized to achieve various tasks including packaging of the workflow and all associated tool specifications. In addition, the existing python library *prov* [296] was used to create a provenance document instance and populate it with the required artefacts generated as the workflow enactment proceeded. This is described in Section 5.4.

5.5.2 *CWLProv* Invocation

CWLProv is built as an optional module to *cwltool* that can be invoked as:

"cwltool −−provenance 'RO-name' wf.cwl job.json"

where "wf.cwl" refers to the workflow file and "job.json" to the input parameter configuration file. This will automatically generate a Research Object with the given name without requiring any additional information from the user. Each input file is named using a checksum and saved in the payload directory *"RO-name/data"*, making it content-addressable to avoid local dependencies.

In order to avoid including attribution information without the consent of the user, an additional flag *" −−enable-user-provenance"* is introduced that is required along with *−−provenance* at the time of invocation. This is given as:

"cwltool −−provenance −−enable-user-provenance 'RO-name' wf.cwl job.json"

If the user provides *−−orcid* and *−−full-name*, this information is recorded as the user attribution details in the provenance profile. Enabling *" −−enable-user-provenance"* and not providing the full name or ORCID will store details of the local machine used as the basis for the attribution, i.e. the details of the *agent* that enacted the workflow.

5.5.3 Process Flow

This process flow of the *CWLProv* RO and the associated provenance profile generation is shown in Figure 5.9. The workflow and command line tool specifications are aggregated in one file to create an executable workflow and subsequently placed in the directory *RO-name/workflow*. This directory also contains

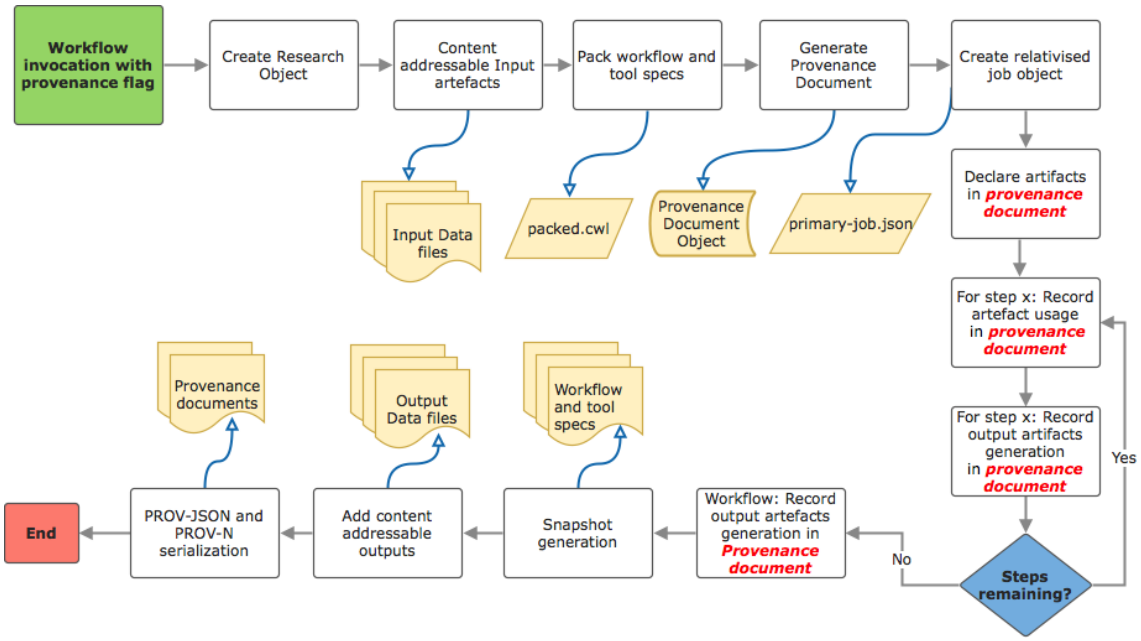

Figure 5.9: High level process flow representation of workflow provenance capture

the modified input job object containing the input parameters with references to artefacts in the *RO-name/data* based on relativising the paths present in the input object. These two files are sufficient to record the input parameters and workflow specifications and can subsequently be used to re-enact the workflow, provided the other required artefacts are also included in the Research Object and comply with the *CWLProv* format. The *cwltool* control flow [297] is used to indicate the points when the execution of the workflow and command line tools involved in the workflow enactment actually start, end and how the output is reported back. This information along with the artefacts are captured and stored in the RO.

When the execution of a workflow begins, the *CWLProv* extensions to *cwltool* generate a provenance document (using the *prov* library). At this point, default namespaces are also added to the provenance document to provide prefixes to the identifiers, e.g. *"prefix data <urn:hash::sha1:>"*. The attribution details are also added at this stage if user provenance capture is enabled, e.g. to answer the ques-

tion "who ran the workflow?". Each step of the workflow can correspond to either a command line tool or another nested workflow referred to as a *sub-workflow* in the CWL documentation. For each nested workflow, a separate provenance profile is initialised (recursively) to achieve *white-box* finer-grained provenance as explained in Section 4.4.3. This profile is continually updated throughout the nested workflow enactment. Each step is identified by a unique identifier and recorded as an *activity* in the parent workflow provenance profile, i.e. the *"primary profile"*. The *nested* workflow is recorded as a step in the *primary profile* using the same identifier as the "nested workflow enactment activity" identifier in the respective provenance profile. For each step in the activity, the start time and association with the workflow activity is created and stored as part of the overall provenance to answer the question "when did it happen?".

The data used as input by these steps is either provided by the user or produced as an intermediate result from the previous steps. In both cases, the *Usage* is recorded in the respective provenance profile using checksums as identifiers to answer the question "what was used?". Non-file input parameters such as strings and integers are stored "as-is" using an additional optional argument, *prov:value*. Upon completion, each step typically generates some data. The provenance profile records the generation of outputs at the step level to record "what was produced?" and "which process produced it?".

Once all steps complete, the workflow outputs are collected and the generation of these outputs at the workflow level are recorded in the provenance profile. Moreover, using the checksum of the files generated by the *cwltool*, content-addressable copies are saved in the directory *"RO-name/data"*. The provenance profile refers to these files using the same checksum such that they are both traceable and can be used for further analysis if required. Finally, the workflow and command line tool specifications are archived in the *snapshot* to preserve the ac-

tual workflow files utilised to enact the workflow.

This prototype implementation of *CWLProv* provides a model and guidance for workflow platforms and executors that can be used to identify features that can be utilised in devising their own provenance implementations.

5.5.4 Achieving recommendations with provenance levels

Table 5.2 map the best practices and recommendations from Table 4.1 to the Levels of Provenance (Figure 4.2). The shown methods and implementation readiness indicate to which extent the recommendations are addressed by the implementation of *CWLProv*.

Note that other approaches may solve this mapping differently. For instance, Nextflow [218] may fulfil ***R19-resource-use*** at *Level 2* (Section 4.4.3) as it can produce trace reports with hardware resource usage per task execution [298], but not for the overall workflow. While a Nextflow trace report is a separate CSV file with implementation-specific columns, our planned *R19-resource-use* approach for CWL is to combine *CWL-metrics* [299], permalinks and the standard *GFD.204* [300] to further relate resource use with *Level 1* (Section 4.4.2) and *Level 2* (Section 4.4.3) provenance within the *CWLProv* Research Object.

In addition to following the recommendations from Table 4.1 through computational methods, the workflow authors are also required to exercise *best practices for workflow design and authoring*. For instance, to achieve ***R1-parameters*** the workflow must be written in such a way that parameters are exposed and documented at workflow level, rather than hard-coded within an underlying Python script. Similarly, while the CWL format support rich details of user annotations that can fulfil ***R6-annotation***, for these to survive into a Research Object at ex-

Table 5.2: Recommendations and provenance levels implemented in *CWLProv*

Recommendation	L0	L1	L2	L3	Methods
R1-parameters	•		•		CWL, BP
R2-automate	•				CWL, Docker
R3-intermediate		•			PROV, RO
R4-sw-version	•		•		CWL, Docker, PROV
R5-data-version	•			•	CWL, BP
R6-annotation		•		*	CWL, RO, BP
R7-described		•			CWL, RO
R8-identifier		•	•	•	RO, CWLProv
R9-environment		*	*		GFD.204
R10-workflow	•	•	•		CWL, wfdesc
R11-software	•		•		CWL, Docker
R12-raw-data	•	•			CWLProv, BP
R13-attribution		•			RO, CWL, BP
R14-provenance		•	•		PROV, RO
R15-diagram	○			*	CWL, RO
R16-open-source	•				CWL, BP
R17-format		•		•	CWL, BP
R18-executable		•			CWL, Docker
R19-resource-use		*	*		CWL, GFD.204
R20-example	*	○			RO, BP

CWL: Common Workflow Language and embedded annotations
RO: Research Object model and BagIt
PROV: W3C Provenance model
CWLProv: Additional attributes in PROV
wfdesc: Prospective provenance in PROV
BP: Best Practice need to be followed manually
• Implemented
○ Partially implemented
* Implementation planned/ongoing

ecution time, such annotation capabilities must actually be used by workflow authors instead of unstructured text files.

It should be a goal of a scientific WMS to guide users towards achieving the required level of the provenance framework through automation where possible. For instance a user may in the workflow have specified a Docker container im-

age without preserving the version, but the provenance log could still record the specific container version used at execution time, achieving ***R4-sw-version*** retrospectively by computation rather than relying on a prospective declaration in the workflow definition.

5.6 Conclusions

Standardisation of common practices through structured objects capturing best practice provenance information can provide more consistent expectations on reproducibility of science. Essential to this process is the interoperable exchange of experimental information supporting cross-platform communication. Coalitions like the Global Alliance for Genomics and Health[13] (GA4GH) are working towards a global standard for interoperable and responsible sharing of both *-omics* and clinical data. Parallel to these efforts, effective sharing of data and computational methods applied to analyse this data is also of paramount importance. In this chapter we defined *CWLProv*, a format for the methodical representation of provenance as part of workflow environment including workflow enactment, capturing associated artefacts, aggregating resources specific to a given enactment and any associated workflow configuration settings. The approach taken focuses on achieving the hierarchical levels of provenance proposed in *Chapter 4* by leveraging existing well-defined, community-driven and agreed upon standards. *CWLProv* benefits from mature efforts such as CWL, Research Object models, BagIt and PROV-DM to resolve different aspects such as workflow definition, method/data stratification/aggregation and provenance representation of workflows. To demonstrate the applicability of *CWLProv*, an existing workflow executor (cwltool) was extended to generate metadata and provenance-rich interop-

[13] https://www.ga4gh.org/

erable workflow-centric ROs, aggregating and preserving data and methods to support the coherent sharing of computational analyses and experiments.

The concept of workflow-centric ROs has been considered by others, e.g. [29, 167, 91, 168, 5], as the basis for structuring the analysis methods and aggregating resources utilised in a given analysis. However, such efforts were largely tied to a single platform, a single WMS or a specific scientific domain. *CWLProv* provides a platform-independent and domain-neutral solution for workflow sharing, enactment and publication. The standards and vocabularies used to design *CWLProv* all have an overarching goal to support domain-neutral and interoperable solutions that can be easily adapted for any domain and shared across different WMS platforms and heterogeneous computing resources. In the next chapter, the interoperability and reproducibility of workflow-centric *CWLProv*-based ROs is evaluated across precisely such a heterogeneous environment using different WMSs.

CHAPTER 6

CWLProv EVALUATION –BIOINFORMATICS WORKFLOW CASE STUDIES

We are at the very beginning of time for the human race. It is not unreasonable that we grapple with problems. There are tens of thousands of years in the future. Our responsibility is to do what we can, learn what we can, improve the solutions and pass them on. –Richard Feynman (1918-1988)

6.1 Introduction

In *Chapter 4* we analysed existing literature focused on defining best practice recommendations for workflow design and subsequent sharing. Drawing on these recommendations, we identified a range of artefacts that directly impact on the provenance of workflows. It was identified that sharing and representation of these artefacts is typically ad hoc due to the myriad approaches used for defining and enacting workflows. In *Chapter 4* we explored how this heterogeneity issue could be addressed by devising a generalised (hierarchical) framework for provenance and associated artefact sharing. We posit that the researchers should highlight the level of provenance they have achieved when publishing an anal-

ysis to help subsequent researchers gauge the magnitude of effort required to understand, reuse and ideally reproduce a given analysis.

Based on the hierarchical provenance framework, we defined *CWLProv*, an interoperable standard and structured format for workflow-centric analysis representation and re-use in *Chapter 5*. The immediate questions is of course, what makes *CWLProv* interoperable? –This must factor in various considerations. Firstly, it must support an underlying workflow definition approach that is specifically designed to achieve portability and interoperability of scientific analyses. This should allow to enact workflows using heterogeneous WMS based on a common declarative and implementation agnostic standard. Secondly, it should support a provenance representation model based on the standards and goals for interoperable exchange of provenance information. It should offer an abstract data aggregation model that is not tied to any specific platform or scientific domain. The choice of standards and methods needs to be considered carefully to precisely support sharing and communication of workflow-centric research across platforms. Importantly, the conceptual modelling of any process in the form of a workflow should be independent of the implementation details, and thereby result in higher level of abstraction that can be optimised by different WMS on different computing platforms whether it be local machines, clouds, clusters or a combination of them.

In this chapter[1], we demonstrate the interoperability of *CWLProv* through an evaluation activity using the reference implementation described in Section 5.5.1 and utilising open source bioinformatics workflows available on GitHub from different research initiatives and from different developers. The evaluation is

[1]Elements of this chapter are submitted in the following article:
Khan, Farah Zaib and Soiland-Reyes, Stian and Sinnott, Richard O. and Lonie, Andrew and Goble, Carole and Crusoe, Michael R. **"Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv".** In: GigaScience (10.5281/zenodo.1966881)

focused on showing how *CWLProv* ROs generated by a CWL executor are reproducible across-WMSs and across different infrastructures and OS, with minimum configuration required on the part of *secondary users* (defined in Section 4.4). In the following sub-sections, we define common use-cases and scenarios that utilise workflow-centric analyses shared by *primary authors* as the basis for the *CWLProv* evaluation. We also discuss concepts commonly associated with interoperability and how these are addressed by *CWLProv*.

6.1.1 Scenarios

In this section, we present three frequently observed scenarios and expectations associated with data-intensive computational research employing computational methods. We identify the *"Goal"* that *secondary users* are interested to achieve. To achieve each of these goals, there is a key *"Requirement"* which has to be satisfied to perform a set of specific *"Actions"*. The term *published research* in the following subsections and in Figure 6.1 refers not only to peer-reviewed publications but also any scientific findings shared with the community.

6.1.1.1 Verify Results

Independent verification of scientific claims and confirmation of the associated published results is crucial to establish trust on any data-driven analysis. This requires *transparent* and *complete* ***communication*** of the research process to be followed to establish and verify the said claims as shown in Figure 6.1 (a). It is the responsibility of the *primary authors* to ensure complete communication of the research process enabling *secondary users* to reproduce the findings published in the original research. This will ultimately provide objective evidence that the

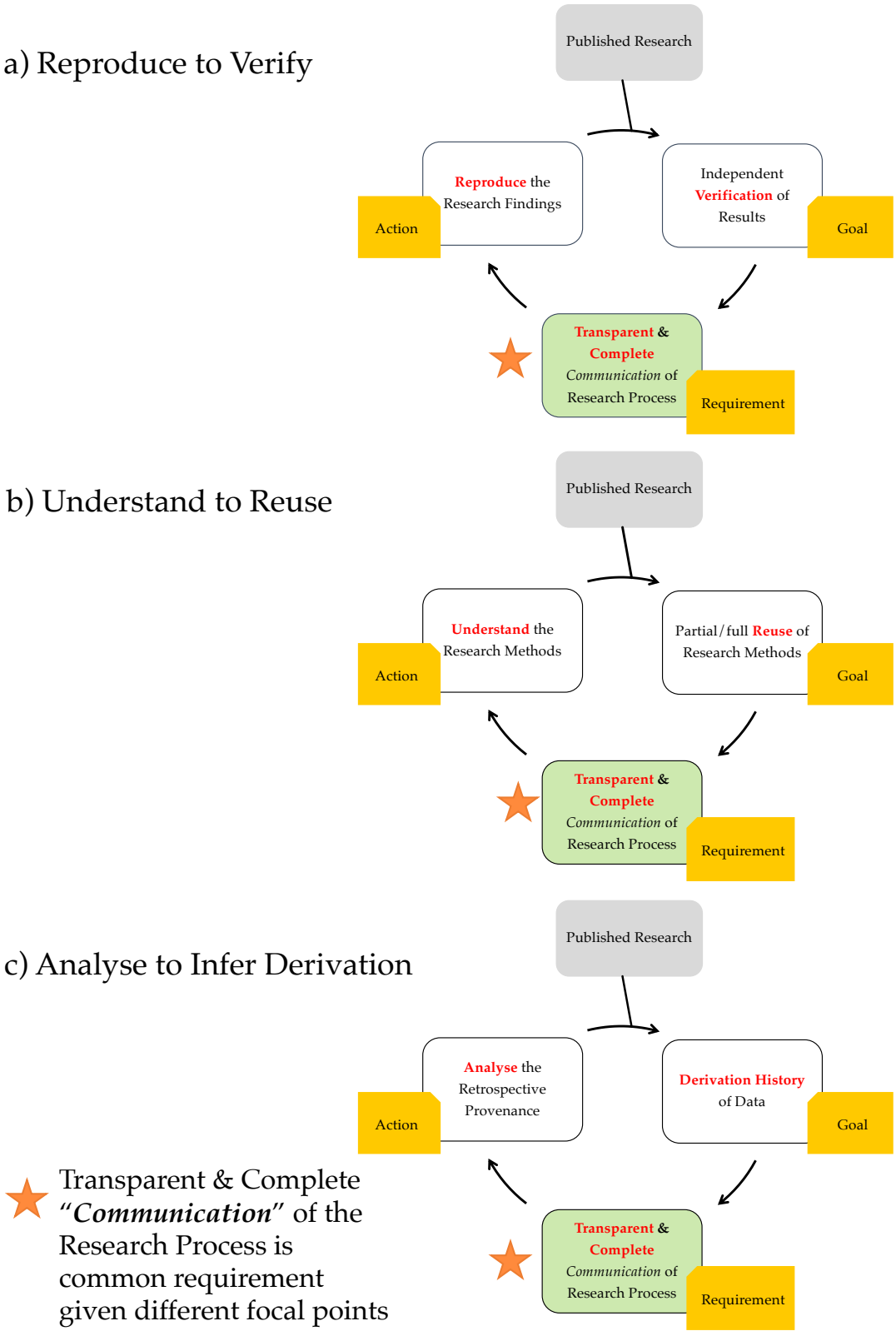

Figure 6.1: Three use-cases focused on the evaluation of *CWLProv* interoperability.

experiment as well as the results are valid.

6.1.1.2 Reuse Methods/Data

Another important benefit of published *in silico* experiments is to build on, re-use and potentially advance the experiments utilising already composed computational methods, cleaned/annotated data and well-tested infrastructure and configuration settings. *Secondary users* may perform a different experiment reusing all or some of the artefacts from the original research published by the *primary authors*. Reusability of methods and data (see Figure 6.1 (b)) can only be achieved if the *secondary users* are able to access and fully understand the original research and where they have domain-specific expertise and knowledge. Hence, the requirement in this scenario is on the *primary users* to ensure transparency and comprehensive communication of the research process in a form that is easy to comprehend.

6.1.1.3 Data Derivation History

Scientific workflows are typically data-flow oriented and provide information of the link between various processing tasks and data artefacts to be utilised to generate a given data artefact (result). To infer factors and artefacts associated with the generation of a given data artefact, the *secondary users* can analyse the workflow definition as well as the retrospective provenance of the workflow enactment provided in the published research by the *primary authors* (see Figure 6.1 (c)). This information can be used to plan for potential partial re-enactments of the targeted tasks without having to re-enact the full workflow to determine the quality and hence the amount of trust one can place on the published findings.

6.1.1.4 Communication Requirements

The three scenarios discussed above have one common requirement, namely the transparent and comprehensive communication of the research process that was followed to obtain the final (published) results. Nowadays, communication in data-driven computational research is increasingly possible since all the artefacts are typically digitised and can be made accessible especially with the Open Source/Open Access movements. Still there is an issue in the lack of conformity in communicating information requiring *secondary users* to invest considerable time and efforts to efficiently use published artefacts. Due to the lack of consensus over the standardised representation of computational analyses including the provenance of results, data and methods, communication is typically not adequately supported. The lack of quality and incompleteness of information communication may render any published experiment ineffective as the *secondary users* fail to understand and reproduce the given analysis.

The communication of the research process in workflow-centric studies should include documentation on: access to the input data, output data and intermediate data artefacts; methods including workflow and tool specifications; aggregated or packaged software dependencies; information on containerised tools, and fine-grained retrospective provenance. All this should be represented in a standardised format to support interoperable workflow objects and enhance reuse of the research artefacts communicated by *primary authors*. This thesis has contributed to the establishment of a format for the standardised communication of the workflow-centric analysis such that the *secondary users* are able to access, understand, analyse and reproduce a given analysis on different WMS and computing platforms. We explore this directly in this chapter.

6.1.2 Interoperability

The concept of interoperability varies in different domains. Here we focus on *computational interoperability*, which is defined as:

"The ability of two or more components or systems to exchange information and to use the information that has been exchanged" [301].

In this chapter we demonstrate how *CWLProv* has achieved *syntactic*, *semantic* and *pragmatic* interoperability as defined in the Levels of Conceptual Interoperability Model (LCIM)[302]. Specifically:

- *Syntactic* interoperability is achieved when a common data format for information exchange is unambiguously defined;
- The next level of interoperability, referred to as *semantic* interoperability, is reached when the content of the actual information exchanged is unambiguously defined, and
- Once there is an agreement about the format and content of the information, *pragmatic* interoperability is achieved when the context, application and use of the shared information and data exchanged is also unambiguously defined.

CWLProv supports *syntactic*, *semantic* and *pragmatic* interoperability of a given workflow and its associated results. We have defined a *"common data format"* for workflow sharing and publication such that any executor or WMS with CWL support can interpret this information and make use of it. This ensures the *syntactic* interoperability between the workflow executors on different computing platforms. Similarly the *"content"* of the shared aggregation artefact as a workflow-centric RO is unambiguously defined, thus ensuring uniform representation of

the workflow and its associated results across different platforms and executors, hence supporting *semantic* interoperability. With *Level 3* provenance satisfied, providing domain-specific information along with level 0-2 provenance tracking as defined in Section 4.4, we posit that *CWLProv* accomplishes *pragmatic* interoperability by providing unambiguous information about the *"context"*, *"application"* and *"use"* of the shared/published workflow-centric ROs. In the next section, widely used bioinformatics workflows are used for the evaluation of *CWLProv* and the experiences and results discussed.

6.2 Bioinformatics Workflows

In this section, we discuss the significance of the exemplar workflows utilised for *CWLProv* evaluation and provide details of the open source CWL specifications provided via GitHub by different (independent) research groups.

6.2.1 RNA-seq Analysis Workflow

RNA sequencing (RNA-seq) data is generated by Next Generation Sequencing (NGS) platforms. RNA-seq is comprised of short sequence reads that can be aligned to a reference genome, where the alignment results form the basis of various analyses such as quantifying transcript expression; identifying novel splice junctions and isoforms, and differential gene expression [303]. RNA-seq experiments can link phenotypes to gene expression and are widely applied in diverse multi-centre studies [144]. Computational analysis of RNA-seq data can be performed by different techniques depending on the research goals and the organism under study [304]. The CWL descriptions of the rna-seq workflow [280] included

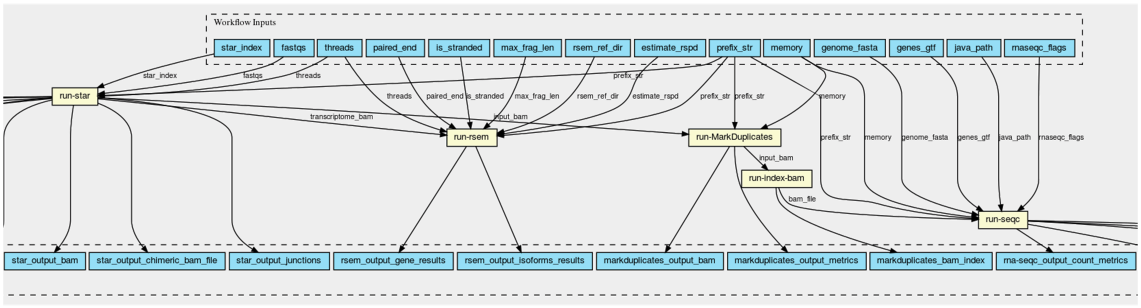

Figure 6.2: Portion of a RNA-seq workflow generated by CWL viewer.

in this case study have been developed by one of the teams [305] participating in the NIH Data Commons initiative [306]. This is a large research infrastructure program aiming to make digital objects such as data generated during biomedical research and software/tools required to utilise such data shareable and accessible, and hence aligned with the **F**indable, **A**ccesible, **I**nteroperable, & **R**eusable (FAIR) principles [307].

The workflow shown in Figure 6.2 was designed for the pilot phase of the NIH Data Commons initiative [308]. It is an adaptation of the approach and parameter settings of the Trans-Omics for Precision Medicine (TOPMed) work [309]. The RNA-seq pipeline originated from the Broad Institute [310]. There are in total five steps in the workflow:

1. Read alignment using STAR [311] that produces aligned BAM files including the Genome BAM and Transcriptome BAM;

2. The Genome BAM file is then processed using Picard MarkDuplicates [312] producing an updated BAM file containing information on duplicate reads (such reads can indicate potential biased interpretations);

3. SAMtools index [313] is then employed to generate an index for the BAM file;

4. The indexed BAM file is further processed with RNA-SeQC [314], which

takes the BAM file, human genome reference sequence and Gene Transfer Format (GTF) file as inputs to generate transcriptome-level expression quantifications and standard quality control metrics;

5. In parallel with the transcript quantification, isoform expression levels are quantified by RSEM [315]. Noting that this step depends only on the output of the STAR tool, and additional RSEM reference sequences.

For testing and analysis, the workflow author provided example data created by down-sampling the read files of a TOPMed public access data set [316]. Chromosome 12 was extracted from the *Homo Sapien Assembly 38* reference sequence and provided by the workflow authors. The required GTF and RSEM reference files were also provided. The workflow was well-documented with a detailed set of instructions of the steps performed to down-sample the data provided. The availability of example input data, use of containers for the underlying software and detailed documentation were important factors in choosing this specific CWL workflow for the *CWLProv* evaluation.

6.2.2 Alignment Workflow

Alignment is an essential step in variant discovery and considered as an obligatory *pre-processing* stage according to Best Practices by the Broad Institute [267]. The purpose of alignment is to filter low-quality reads before variant calling or other interpretative steps [317]. A workflow for alignment is typically designed to operate on raw sequence data to produce analysis-ready BAM files as output. The typical steps followed include file format conversions, aligning the read files to the reference genome sequence, and sorting the resulting files.

The CWL definition of the alignment workflow [318] included in this evalu-

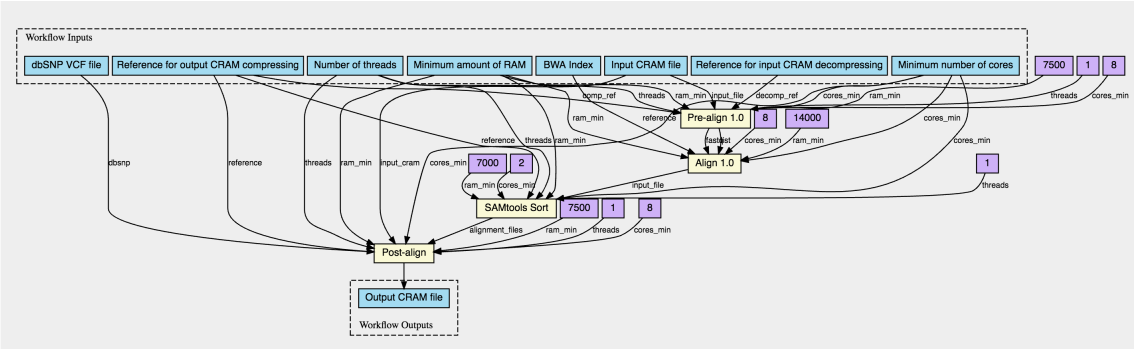

Figure 6.3: Alignment workflow representation generated by the CWL viewer.

ation (see Figure 6.3) was designed by Data Biosphere [319]. It adapts the alignment pipeline [320] originally originally developed at Abecasis Lab, The University of Michigan [321]. This workflow is also part of NIH Data Commons initiative (as the RNA-seq workflow described in Section 6.2.1). It comprises the following four stages:

1. First step, "Pre-align" accepts a Compressed Alignment Map (CRAM) file (a compressed format for BAM files developed by European Bioinformatics Institute (EBI) [322]) and human genome reference sequence as input and using the underlying software utilities from SAMtools such as view, sort and fixmate, it returns a list of FASTQ files which are to be used as input for the next step.

2. The next step "Align" also accepts the human reference genome as input along with the output files from "Pre-align" and uses BWA-mem [230] to generate aligned reads. SAMBLASTER [323] is used to mark duplicate reads and SAMtools view to convert read files from SAM to BAM format.

3. The BAM files generated after "Align" are sorted with "SAMtool sort".

4. Finally these sorted alignment files are merged to produce a single sorted BAM file using SAMtools merge in the "Post-align" step.

The authors provide a custom script[2] to download the files from the Google Cloud including an example CRAM file, the *Homo Sapien Assembly 38* reference genome along with the index files to be used as inputs for testing and analysis of the workflow. This requires installation of gsutil[3], a Python application used to access the Google Cloud storage and hence to download the data. This step is not required when reproducing this evaluation study since we aggregated data in the *CWLProv* RO as discussed in Section 6.3.

6.2.3 Somatic Variant Calling Workflow

Variant discovery analysis for high-throughput sequencing data is a widely used bioinformatics technique. It focuses on finding genetic associations of diseases including identifying somatic mutations in cancer and characterising heterogeneous cell populations [324]. The *pre-processing* explained for the Alignment workflow can be part of any variant calling workflow since reads are classified and ordered as part of the variant discovery process. Numerous variant calling algorithms have been developed depending on the input data characteristics and the specific application areas [317]. Somatic variant calling workflows are designed to identify somatic (non-inherited) variants in a sample, e.g. a cancer sample, by comparing the set of variants present in a sequenced tumour genome with a non-tumour genome from the same host [325]. The set of tumour variants is a super-set of the set of host variants, and somatic mutations are identified through various algorithmic approaches to subtracting host familial variants. Each somatic variant calling workflow typically consists of three stages: pre-processing; variant evaluation and post-filtering.

[2]https://github.com/DailyDreaming/fetch_gs_frm_json
[3]https://cloud.google.com/storage/docs/gsutil

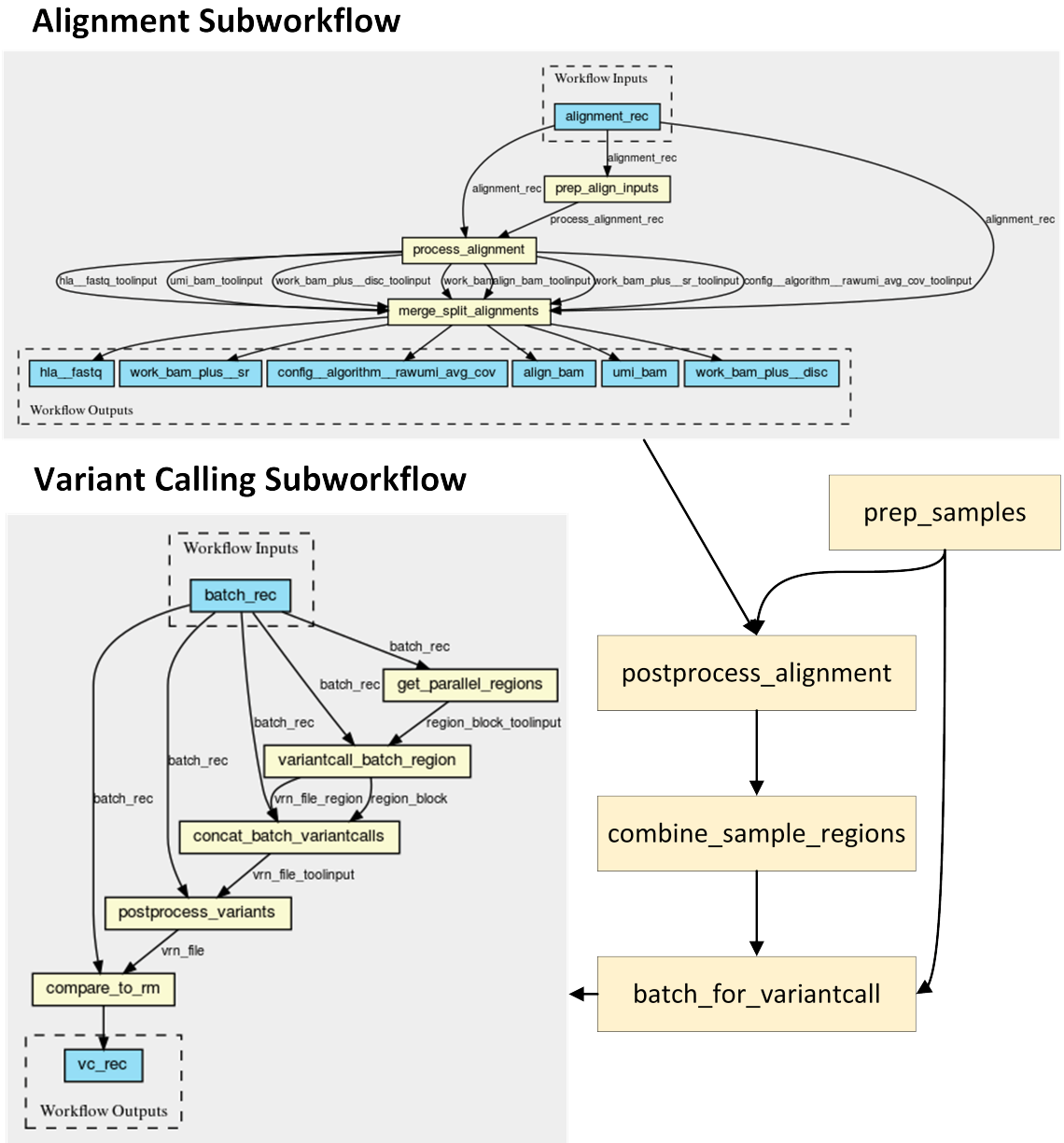

Figure 6.4: Visual representation of the bcbio somatic variant calling workflow adapted from [328] with subworkflow images generated by the CWL viewer.

The somatic variant calling workflow (see Figure 6.4) included in this case study was designed by Blue Collar Bioinformatics (bcbio) [326], a community-driven initiative to develop best-practice pipelines for variant calling of RNA-seq and small RNA analysis. According to the documentation, the goal of this project is to facilitate the automated analysis of high throughput data by making resources *quantifiable*, *analysable*, *scalable*, *accessible* and *reproducible*. All of the underlying tools are containerized. The somatic variant calling workflow defined in CWL is available on GitHub [327] and equipped with a well defined test dataset.

6.3 *CWLProv* Evaluation Activity

This section describes the evaluation of the cross-executor and cross-platform interoperability of *CWLProv* . To test cross-executor interoperability, two CWL executors *cwltool* and *toil-cwl-runner* [217] were selected. *toil-cwl-runner* is an open source Python workflow engine supporting robust cross-platform workflow execution on Cloud and High Performance Computing (HPC) environments. The two operating system platforms utilised in this analysis were MacOS and Ubuntu Linux. For the Linux OS, a 16-core Linux instance with 64GB RAM was launched on the NeCTAR research cloud. To cater for the storage requirements, a 1000GB persistent volume was attached to this instance.

For MacOS, a local system with 16GB RAM, 250GB storage and 2.8 GHz Intel Core i7 processor was used. These platforms were selected to cater for the required storage and compute resources of the workflows described above. The reference genome provided with the Alignment workflow (Section 6.2.2) was not down-sampled and hence this workflow required the most resources among the three workflows evaluated. It is to be noted that this evaluation does not include details of the installation process for *cwltool*, *toil-cwl-runner* and *Docker* on the sys-

tems described above. To create *CWLProv* ROs during workflow execution, it is necessary to use the CWL reference runner (*cwltool*) until this practice spreads to other CWL implementations. Moreover, it is assumed that the software container (Docker) is installed on the system to use the workflow definitions aggregated in a given *CWLProv* RO.

In addition, the resource requirements (discussed in Section 5.2.1.2) should also be satisfied by choosing a system with enough compute and storage resources for enactment. The systems used in this case study provide a reference that can be used when selecting a system since inadequate compute and storage resources will impact on the successful re-enactment of workflows using these ROs. The hardware requirements may also vary if a different dataset is used as input to re-enact the workflow using the methods aggregated in the RO. In that case, the end user must ensure the availability of adequate compute and storage resources by choosing a system that meets the required specifications.

Since the *CWLProv* implementation is demonstrated for one of the executors (*cwltool*), currently a *CWLProv* RO for any workflow can only be produced using *cwltool*. Hence, in this activity the workflows are initially enacted using just *cwltool* (Table 6.1), although the re-enactment using these Research Objects can be undertaken using other CWL executors. The steps performed to analyse *CWLProv* each case study is as follows:

1. The workflow was enacted using *cwltool* to produce a RO on a MacOS computer.
    (a) The resulting RO and aggregated resources were used to re-enact the workflow using *toil-cwl-runner* on the same MacOS computer;
    (b) The RO produced in step 1 was transferred to a cloud-based Linux instance used in this activity;

Table 6.1: *CWLProv* evaluation summary and status for the bioinformatics case studies.

Enact-produce RO with	Re-enact using RO with	Status
cwltool on MacOS	*toil-cwl-runner* on MacOS	✓
	cwltool on Linux	✓
	toil-cwl-runner on Linux	✓
cwltool on Linux	*toil-cwl-runner* on Linux	✓
	cwltool on MacOS	✓
	toil-cwl-runner on MacOS	✓

(c) On the cloud-based Linux environment and only utilising the resources aggregated in the RO, the workflow was successfully re-enacted using both *cwltool* and *toil-cwl-runner*.

2. The workflow was enacted using *cwltool* to produce a Research Object on Linux.

   (a) The resulting Research Object and aggregated resource were utilised to re-enact the workflow using *toil-cwl-runner* on the same cloud-based Linux instance;

   (b) The Research Object produced in step 2 was transferred to the MacOS computer used in this activity;

   (c) On the MacOS computer and only utilising the resources aggregated in the Research Object, the workflow was successfully re-enacted using *cwltool* and *toil-cwl-runner*.

The *CWLProv* ROs produced as a results of this activity are published on Mendeley Data [329, 330, 331].

6.4 Evaluation Results

The steps described above were taken to produce Research Objects which were then used to (successfully) re-enact the workflows, without any further changes required. This demonstration illustrates the syntactic, semantic and pragmatic interoperability of the workflows across different executors and operating systems. It shows that **both CWL executors were able to *exchange*, *comprehend* and *use* the information represented as *CWLProv* ROs**. One of the scenarios, verification of results discussed in Section 6.1.1.1, is satisfied primarily as the information communicated was utilised and the experiment reproduced. In the following subsections we discuss different aspects of this evaluation.

6.4.1 *CWLProv* and Interoperability

CWL already builds on technologies such as JSON-LD for data modelling and Docker to support portability of run-time environments. Portability and interoperability as basic principles of the underlying workflow definition approach for any workflow-centric analysis implies (obviously!) that the analysis should also be portable and interoperable. However, the workflow definition/specification alone is insufficient when dealing with command line tool specifications, data, and input configuration files used in the analysis.

CWLProv ensures availability of these resources for a given analysis conforming to the framework defined in Section 5.3. The input configurations are saved as *primary-job.json* in directory *workflow/* and refer to the input data contained in the payload *data/* directory of the given RO. In this way, availability of data aggregated with the analysis is made possible. Existing features of *cwltool* are used to generate the CWL workflow specification file containing all of the command line

tool specifications referred to in the workflow specification and these are placed in the same *workflow/* directory.

One might argue that copying a directory tree might serve the same purpose but in that case it is necessary to rely on users to put a substantial amount of effort on top of the actual analysis, i.e. they would have to carefully structure their directories to be aligned with the workflow creators. Instead CWL encourages researchers to utilise container technologies such as Docker, Singularity, Debian (Med) or Bioconda packages to ensure the availability of the underlying tools as recommended by numerous studies (Table 4.1). This practice facilitates the preservation of methods utilised in data-intensive scientific workflows and enables verification of published claims without requiring the end-user to perform any manual installation and configuration. Examples of the tools made available in Docker containers here include the alignment tool (BWA mem) used in the Alignment workflow and STAR aligner used in the RNA-seq workflow.

6.4.2 Evaluating Provenance Profile

The retrospective provenance profile generated as part of *CWLProv* for each workflow enactment can be examined and queried to extract the required subset of information. *Provenance Analytics* is a separate domain and a next step following provenance data collection in the provenance life cycle [30]. Often provenance data is queried using specialised query languages such as SQL, SPARQL or TriQL depending on the storage mechanism used. Query operations can combine information from prospective and retrospective provenance to better understand computational experiments. Hence, the provenance profile generated and communicated as part of *CWLProv* RO when queried, helps in understanding the derivation history of a given data product as discussed in Section 6.1.1.3.

The focus of this thesis is not in-depth provenance analytics but we have demonstrated the application of the provenance profile generated as part of *CWLProv*. We have developed a command line tool and Python API *"cwlprov-py"* [332] for *CWLProv* RO analytics to interpret the captured retrospective provenance of CWL workflow enactment. This API currently supports several use cases as described below.

6.4.2.1 Workflow Runs

Each *CWLProv* RO can contain more than one *workflow run* if sub-workflows are designed to group related tasks into one workflow. In that case, the provenance traces are stored in separate files for each workflow run. *cwlprov-py* identifies the workflow enactments including the sub-workflows *(if any)* and returns the workflow identifiers annotated with the step names. Users can subsequently select the required provenance trace and explore it in detail.

6.4.2.2 Attribution

Each *CWLProv* RO is assumed to be associated with a single enactment of the primary workflow and hence assumed to be enacted by one person. As discussed previously, *CWLProv* implementation provides additional flags to enable user provenance capture. A user can provide their name and ORCID details that can be stored as part of the Research Object. *cwlprov-py* displays attribution details of the researcher responsible for the enactment *(if enabled)* and the version of the workflow executor utilised in the analysis.

6.4.2.3 Input/Output of a Process

Provenance traces contain associations between the steps/workflows with the data they use as input or generate as output. A user interested in a particular step of a workflow enactment can use *cwlprov-py* to identify and view the inputs used and outputs produced and how they are linked explicitly to a given process.

6.4.2.4 Partial Re-runs

Re-running or re-using only parts of a given workflow has been emphasised [144] as important to evaluate the workflow process or validate the published results associated without necessarily re-enacting the workflow as a whole. *cwlprov-py* uses the identifier of the step/workflow to be re-run, parses the provenance trace to identify the inputs required and subsequently creates a JSON input object with the associated input parameters. This input object can be used for partial re-runs of the desired step/workflow making segmented analysis possible without unnecessary computations. This demonstrates how provenance information if communicated along with other resources such as workflow specifications can help in partial/full reuse of the published research as discussed in Section 6.1.1.2.

6.4.3 Temporal and Spatial Overhead with Provenance

Table 6.2 shows the run-times for the three workflow enactments using both cwl-tool and toil-cwl-runner on Linux and MacOS platforms with and without enabling provenance capture as described in the evaluation activity section. These workflows were enacted at least once before this time calculation, hence the timing does not include the time for Docker images to be downloaded. On a new

Table 6.2: Run-time comparison for the workflow enactments done cross-executor and cross-platform

	Linux			**MacOS**		
Workflow	**cwltool**		**toil-cwl-runner**	**cwltool**		**toil-cwl-runner**
	With Prov	W/O Prov	W/O Prov	With Prov	W/O Prov	W/O Prov
RNA-Seq Analysis Workflow	4m30.289s	4m0.139s	3m46.817s	3m33.306s	3m41.166s	3m30.406s
Alignment Workflow	28m23.792s	24m12.404s	15m3.539s	–	162m35.111s	146m27.592s
Somatic Variant Calling Workflow	21m25.868s	19m27.519s	7m10.470s	17m26.722s	17m0.227s	**

** This could not be tested because of a known bug: https://github.com/common-workflow-language/cwltool/issues/328
– This could not be tested because of the insufficient hardware resources on the MacOS

system, when re-running these workflows for the first time, the Docker images need to be downloaded which may take significantly longer than the time specified here especially in case of the Somatic Variant Calling workflow because of the Docker image sizes.

Run-time and storage overheads are important for provenance-enabled computational experiments. The choice of operating system and the provenance capture mechanisms such as the operating-system level, application-level or workflow-level as well as the I/O workload, interception mechanism and fine-grained information capture are all key for provenance [333, 334].

It is noted that the time differences between the cwltool and toil-cwl-runner enactments are due to the default parallel versus serial job execution in the case of *toil-cwl-runner* and *cwltool* respectively. The "scatter" operation in CWL when applied to one or more input parameters of a workflow step or a sub-workflow, supports parallel execution of the associated processes. Parallelism is also available without scatter when separate processes have all of their inputs ready. If sufficient compute resources are available, these jobs will be enacted concurrently otherwise they are queued for subsequent execution. Compute intensive steps of a workflow can benefit from scatter features for parallel execution by reducing the overall run-time. Both Alignment and Somatic Variant Calling workflows utilise the scatter feature to enable higher degrees of parallel job execution in the case of *toil-cwl-runner* which explains the time difference for the cross-executor of these two workflows. The difference is negligible for RNA-Seq workflow however, since it is comprised of serial jobs with comparatively small test data.

6.4.4 Output Comparison Across Enactments

We compared the workflow outputs after each enactment to observe the concordance and/or discordance (if any) of the workflow enactment results produced across the platforms and across the executors. As *CWLProv* RO refers to data with hashed checksums, these checksums are utilised for result comparison. The comparison was made between the output files generated by the different enactments against a single *"truth-set"* output file and checksum available in the respective Git repositories.

The checksum of the output data generated for the cross-platform and cross-executor comparison data as a result of the initial enactment and re-run using the *CWLProv* ROs was concordant in all but one cases. Specifically, the "correctness" and agreement of the outputs given different execution environments (e.g. platform and executor) held true with the exception of the Alignment workflow. The Alignment workflow produced varying outputs after every execution even with the same executor and platform. The output of the alignment algorithm, "BWA mem" used in this workflow is a non-deterministic algorithm as depends on the *number of threads –t* and the *seed length –K* that can affect the output produced. While the seed length in this case was set to a constant value, the number of threads varied depending on the availability of hardware resources at run-time, hence resulting in varying output for the same input files.

6.5 Discussion

This section discusses the ongoing and future work with reference to enriched provenance capture and the associated enhancements to both the *CWLProv* standard and its implementation.

6.5.1 Compute and Storage Resources

The *CWLProv* format encapsulates the data and workflow definitions involved in a given workflow enactment along with its retrospective provenance trace. As discussed in Section 5.2.1.2, CWL as a standard provides constructs to declare basic hardware resource requirements required for a particular workflow enactment and workflow authors can encouraged to provide this information in the *"requirements"* section of CWL as a *"ResourceRequirement"*. These requirements can be declared at the workflow or individual step level, to guide platforms/executors in the establishment of the required resources. This information indirectly stores some aspects of prospective provenance with respect to the hardware requirements of the underlying system used to enact the workflow. Currently this information is only available if formally declared as part of the workflow specification. In future, we plan to include these requirements as part of provenance for a given workflow such that all such information is gathered in one space and users are not required to inspect multiple sources to extract this information. This information can subsequently be used as a pre-condition for the successful enactment of a given workflow.

As *CWLProv* is focused on retrospective provenance capture of workflow enactment, we plan to include provenance information about the compute and storage resources utilised in a given enactment to fulfil ***R19-resource-use*** of Table 4.1. Documenting these resources will allow users to analyse their environment and resource allocations before execution, as opposed to trial and error methods that may result in multiple failed enactments of a given workflow. Despite the obvious importance, most existing provenance standards lack dedicated constructs to represent underlying hardware resource usage information as part of prospective or retrospective provenance. In the case of complex workflows using distributed resources, where each step can be executed on a different node/server, including

all this information in a single *PROV* profile may clutter the profile and render it potentially incomprehensible. Therefore, creation of a separate *Usage Record* document in the *CWLProv* RO conforming to GFD.204 [300] to describe *Level 1* (Section 4.4.2) and potentially *Level 2* (Section 4.4.3) resource usage in a common format that is independent of the actual execution environment would be beneficial.

Capturing such resource usage records requires a tighter integration with the execution platform, that is only available in tools such as *Toil* or *Arvados*. At present the reference implementation *cwltool* does not exercise fine-grained control for task execution. Detailed raw log files can also be provided as *Level 0* provenance, as demonstrated with cwltool, but these will by their nature be bespoke and customised per execution platform and thus remain non-standard. Related work that is already exploring this approach is *cwl-metrics* [299], which analyses raw *cwltool* log files in combination with detailed Docker invocation statistics using the container monitoring tool *Telegraf*[4]. Ongoing collaboration with that group is exploring adding these metrics as additional provenance to the *CWLProv* research object with summaries in PROV and GFD.204 formats.

6.5.2 Provenance Profile Augmented with Domain Knowledge

CWLProv benefits from leveraging best practices proposed by numerous studies (Table 4.1) including use of standards for workflow representation, resource aggregation and provenance tracking (Section 5.2). We posit that the principle of following well-defined data and metadata standards enables explicit data sharing and reuse. In order to include rich metadata for bioinformaticians to produce specialised Research Objects for bioinformatics to achieve *CWLProv Level 3* as de-

[4]https://www.influxdata.com/time-series-platform/telegraf/

fined in Section 4.4, we are investigating re-use of concepts from the BioCompute Object (BCO) project [169]. This domain-specific information is not necessary for computation and execution but for understandability of the shared resources.

We encourage workflow authors to include such metadata and external identifiers for data and underlying tools, e.g. EDAM identifiers. These annotations will be extracted and represented as part of the retrospective provenance profile in *CWLProv*. Domain-specific information is essential in determining the nature of inputs, outputs and context of the processes linked to a given workflow enactment [269]. This information can be captured in a *CWLProv* RO if and only if the workflow author adds it in the workflow definition. Hence achieving *CWLProv Level 3* depends on the individual workflows and workflow authors.

6.5.3 Big *-omics* Data

While aggregating all resources as one download-able object improves reproducibility, the size of the resulting Research Object is an important factor in practice. On the one hand, completeness of the resources contributes towards minimising the *workflow decay* phenomenon by minimising the dependence on the availability of third party resources. On the other hand, the nature of *-omics* data sizes can result in hard-to-manage workflow-centric Research Objects that leads to considerable overheads.

One solution is archiving big datasets in online repositories or data stores and including persistent identifiers and checksums in the Research Object instead of including the actual data files, as previously demonstrated with BDBags [283]. While CWL executors like *toil-cwl-runner* can be configured to deposit data in a shared repository, the *cwltool* reference implementation explored in this study can

only write to the local file system. External references raise the risk of unavailability of data at a later time. Therefore we recommend including the data in the Research Object if sufficient network and storage resources are available. Future work may explore post-processing *CWLProv* ROs to replace large data files with references to stable data repositories, producing a reduced size Research Object for transfer where individual data items can be retrieved on demand, as well as reducing data duplication across multiple related Research Objects.

6.5.4 Customised *CWLProv* RO with Selected Jobs Provenance

Workflows may have necessary intermediate steps in which data are large, but not particularly interesting, such as converting a tab-separated file to a comma-separated file. Provenance data from such steps can greatly increase the storage cost with little information gain. Such data can often be recreated by re-applying a given transformation step. Future *CWLProv* work could add options to ignore capturing outputs of specified *shim* steps, or ignoring files over a particular file size [244]. Shim steps employ tools to convert the output of the previous step into an acceptable format for the next step in a workflow. For example in our case study RNA-seq workflow, RNA-SeQC required an indexed BAM file whilst the output of STAR or Picard MarkDuplicates is only comprised of the BAM file. Hence, the SAMtools index was utilised afterwards to make the aligned reads analysis ready for RNA-SeQC. A WMS may also only capture provenance at a particular provenance level (see Section 4.4). It is envisioned that such partial provenance capture can be indicated in the RO manifest as a variant of the *CWLProv* profile identifier to give the end-user a clear indication of what to expect in terms of completeness.

6.6 Enforcement of Best Practices –An Open Problem

Recommendations and best practices from the scientific community are proposed frequently to guide researchers to design their computational experiments in such a way as to make their research verifiable and reusable. This can include best practices for workflow design as well as for resource declaration, software packaging and configuration management [156] to avoid dependencies on local installations and minimise manual processes, e.g. for dependency management. The term *"Better Software, Better Research"* [335] applies directly to workflows and the workflow design process.

Declarative approaches to workflow definition such as CWL encourage users to explicitly declare everything in a workflow, improving the white-boxness of retrospective and prospective provenance. Such workflows should ideally transparently communicate the complete research process used for producing given data artefacts. However, it is entirely up to researchers to implement approaches to produce well-defined workflows with explicit details capturing provenance traces at the appropriate level. This currently requires considerable effort on the workflow designer (*primary author's*) behalf.

As one example, the alignment workflow used in this case study (Section 6.2.2) embeds bash scripts and complex JavaScript expressions into the CWL command line tool definition, requiring another layer needed to be explored for provenance information extraction. Specifically it combines multiple underlying tools such as BWA-mem, SAMBLASTER and SAMtools view into one command line tool description document used for the step "Align". This approach is similar to most pre-built pipelines and GUI based systems where the tasks are masked into one activity for simplified user view as discussed in Section 3.3.2. These three underlying tools perform different functions however and should be treated as

three independent steps with separate CWL command line tool descriptions.

Despite using CWL for workflow definition and *CWLProv* for provenance capture of the alignment workflow, the declarative constructs discussed in Section 5.2.1.2 are not fully utilised as each underlying tool is not explicitly declared as an independent step. This results in a provenance profile missing critical information making it coarse-grained. The raw logs capturing the enactment are also not always informative in these cases. These factors hinder the transparent communication and complete provenance capture of the individual steps performed and the parameter settings that are utilised.

The three criteria defined by Cohen *et al.* [144] to be followed by workflow designers include: modularized specifications, unified representation and workflow annotations. CWL facilitates a modular structure to workflow definitions by coupling similar steps to *subworkflows*. As an interoperable standard, it provides a common platform moving towards resolution of the heterogeneity issues of workflow specification languages. In addition, users can add domain-specific annotations to data and workflows to enhance understanding of the shared specification. All these features can be utilised by workflow designers to produce better workflows and maximise the information declaration resulting in semantically-rich and provenance-complete *CWLProv* ROs.

Workflow-centric initiatives similar to *software carpentry* [336] and *code is science* [337] are one possible way to organise training and create awareness around such best practices. Community-driven efforts to consolidate the understanding of requirements to make a given workflow explicit, understandable and reproducible should be made. Furthermore, not only awareness about the workflow design is needed, but also the availability of the associated resources should be emphasised e.g. software containers or software packages, big datasets in public repositories and pre-processing/post-processing that needs to be included as

part of a given workflow. Without putting such proposed best practices into actual practice, the complete communication and hence the reproducibility of workflow-centric computational analyses is likely to remain challenging.

Chapter 7

Conclusions & Future Work

True science thrives best in glass houses where everyone can look in. When the windows are blacked out, as in war, the weeds take over; when secrecy muffles criticism, charlatans and cranks flourish –Dr Max Perutz (1914-2002)

Presenting science in glass houses instead of ivory towers depends on transparent communication on the researcher/authors' part and rigorous scrutiny from the audience comprising the scientific community, funding bodies, policy makers, journal reviewers to name a few. Provenance is an essential criterion to bring transparency to scientific investigations and establish trust on the reporting of research results. When documented methodically, provenance should facilitate understanding of the research process followed to reproduce any novel finding. The increased adoption of sophisticated computational methods in data-intensive experiments such as *-omics* analysis, where most of the artefacts are linked with experiment are digitised and can/should be disseminated when findings are published. It is important that these artefacts can be utilised for verification of the associated findings or re-used fully or partially as supporting tools for studies adopting the same methods or with the same overarching goals.

The goal of this thesis was to further the understanding of the fundamental elements of bioinformatics workflow provenance and develop mechanisms

for the standardised representation of workflow-centric studies and tackle the issues with transparent communication of provenance information and its reuse. The objectives and the research outcomes of this thesis align well with the FAIR principles [307] with respect to workflows and especially the Interoperability and Reusability of *in silico* workflow-centric components. The *R* in FAIR is also referred to as Reproducibility[1] which is also considered as a crucial provenance application of this thesis. During the process of empirical and pragmatic analysis carried out during this research, we extracted several overarching conclusions that can apply to any scientific workflow-centric study irrespective of the research domain.

As Tim Peters[2] says *"explicit is better than implicit"*, the same rule applies to workflow definitions. Incomplete or coarse-grained provenance is often caused by implicit assumptions considered needless to be stated during the workflow design, however these can directly impact on future workflow enactments. These intricate details if not captured and communicated via provenance when publishing a workflow analysis directly influence the transparency of any computational methodology. Therefore, approaches to workflow definition should be built on the principles of explicit declaration of the requirements of each analytical step required for workflow design to ensure comprehensive provenance documentation.

A second conclusion from this work is that *workflow abstraction can address the heterogeneity of workflow specifications to improve their interoperability across different WMS and computing platforms*. Ideally, an abstract but *executable* workflow representation guided by strict design principles extracted from well-defined and widely adopted standards can be optimised and enacted by any supporting WMS irrespective of the underpinning computing platform. Such workflow specifica-

[1]https://www.slideshare.net/carolegoble/being-fair-enabling-reproducible-data-science
[2]https://github.com/python/peps/blob/master/pep-0020.txt

tions should contain references to any/all software dependencies required to be satisfied including the lower level dependencies, e.g. at the container, configuration or package manager levels. CWL is an exemplar of one such standardised approach that is termed as the *"future trend"* [175] and *"lingua franca"* [144] for large-scale workflow design that enables portability of workflow analyses. Leading WMSs are moving towards and supporting CWL. Examples for bioinformatics analysis platforms adopting CWL include Galaxy and Taverna (implementation currently in progress). Examples of platforms delivering end-to-end bioinformatics solutions adopting CWL include Seven Bridges[3] and Arvados[4]. Examples of distributed-computing oriented workflow engines such as Toil are either already supporting enactment of CWL workflow specifications or making significant progress in this respect. We have demonstrated in the previous chapter how adopting a workflow definition approach supports sophisticated abstraction levels supporting syntactic, semantic and pragmatic interoperability of the workflows.

A third conclusion discussed in Section 6.6 was based on an example from practical experiences. Specifically, it was identified that an immediate need exists for initiatives focused on creating awareness about best practices associated with workflow design processes to improve provenance documentation. Provenance capture and its subsequent use to support published research transparency and integrity should not be treated as an after-thought but rather as a standard practice of up-most priority. As evident from literature reviewed during this research, the assumption of black-box provenance is often associated with the workflows and used to justify the coarse-grained provenance of workflow steps. This makes the finer-grained traceability between independent tasks, their parameters and results difficult/impossible. It is argued that with well-defined

[3]https://www.sevenbridges.com/
[4]https://arvados.org/

standards for provenance and declarative workflow definition approaches, this assumption can be addressed by adopting best practices put forward by the scientific community. Despite the use of declarative approaches, users still might not explicitly differentiate between independent tasks leading to black-box representation of provenance. A simple example of implicit information can be seen in case of the workflow described in Section 6.2.2, where a single workflow step was used for performing three independent tasks by invoking three underlying tools. This clearly requires the implementation of a more modular solution associated with these tasks resulting in a sub-workflow. We do not require new standards, new WMSs or indeed new best practices, instead the focus should be to implement, utilise and re-use existing community-driven initiatives handling different aspects of computational experiments. The remainder of this chapter summarises the contributions of this thesis and the implications of the research carried out along with its limitations and future directions to be explored.

7.1 Contributions

In this section we enumerate the contributions made in this thesis and describe how these contributions address the research questions defined in Section 1.3 by meeting the objectives defined in 1.2 that are summarised in Table 7.1.

To answer the research question **RQ1**: ***"How do existing WMSs handle different aspects of governance?"***, we have made the following contribution.

C1.1: Taxonomy Classification

The provenance taxonomy devised in this thesis (presented in *Chapter 2*), focuses on provenance aspects specific to scientific workflows that are to develop deeper understanding of workflow provenance and the supporting

resources that are required to fully utilise the captured provenance information. To fully utilise this provenance information, this taxonomy improves understanding of the supporting artefacts and can serve as a guide when any workflow definition approach or WMS is evaluated. This contribution partially meets our objective *"To investigate how various workflow definition approaches address the key aspects of provenance"* **(**O1).

C1.2: Workflow Definition Approaches Classification

Due to the myriad workflow definition and implementation approaches, and the variety of platforms available for genomic data analysis, a categorisation was necessary for informed provenance analysis. We focused on several leading exemplar systems representing different approaches. Specifically, we classified existing workflow definition approaches into three broad categories: Domain-specific Pre-built Pipelines, Graphical User Interface (GUI)-based Integrative Workbenches and Standardised Approach to Workflow Definition. This categorisation identified the commonality and uniqueness of the different systems. This contribution along with **C1.1** meets our objective *"To investigate how various workflow definition approaches address the key aspects of provenance"* **(**O1).

To answer the research question **RQ2**: ***"What are the key artefacts that must be documented for comprehensive provenance capture in bioinformatics workflows?"***, we have made the following three contributions.

C2: Identification of Implicit Assumptions

Through the empirical study implementing a complex yet widely used variant calling workflow on an exemplar system using each category of workflow definition approaches devised in C1.2, we identified a range of underlying implicit assumptions of each approach. These assumptions lead to

Table 7.1: Contributions made in this thesis to answer the Research Questions

RQs	**Contributions**	**Presented In**	Meets Objectives
RQ1	**C1.1**, **C1.2**	Ch2	O1
RQ2	**C2**, **C3**, **C4**	Ch3, Ch4	O2, O3, O4
RQ3	**C4**, **C5**	Ch4	O5
RQ4	**C6**, **C7**, **C8**, **C9**	Ch5, Ch6	O6, O7

missing details and documentation of workflow enactment provenance resulting in coarse-grained provenance. Understanding the implications and subsequent impact of such assumptions and hence avoiding them leads to finer-grained provenance documentation including documenting the artefacts required for transparent communication of the workflow analysis. This contribution detailed in *Chapter 3* meets our objective: *"To identify implicit and explicit assumptions of workflow definition approaches that lead to incomplete provenance documentation"* **(**O2).

C3: Comprehensive Recommendations

We proposed a set of recommendations to mitigate the assumptions identified through **C2**. These recommendations guide the scientific community on the choice of workflow framework, licensing and generally open-science practices that can be used to improve the capture of fine-grained retrospective provenance to address a key application of provenance information, *reproducibility*. This contribution detailed in *Chapter 3* extends the work done to meet our objective: *"To identify implicit and explicit assumptions of workflow definition approaches that lead to incomplete provenance documentation"* **(**O2) by providing a generalised but comprehensive set of recommendations identified through the empirical study.

C4: Identification & Understanding of Provenance Artefacts

This contribution was made with the help of empirical case study undertaken for **C2** and **C3** (*Chapter 3*) as well as through the literature classification (*Chapter 4*). The assumptions in **C2** are characterised to identify missing artefacts from workflow provenance and uninformed workflow design practices that lead to incomplete workflow provenance documentation. The identification of these artefacts impacting the provenance granularity improved the understanding of crucial factors that must be considered when designing, enacting and sharing a workflow and its associated analysis. The details of these artefacts are presented in *Chapter 3*. This contribution meets our objective: *"To develop an understanding of the essential factors/artefacts/resources that must be captured as part of provenance in light of the identified assumptions in O2"* **(**O3).

Instead of basing further research on a single empirical study, the understanding of such factors was further extended to include the experiences of the scientific community. The recommendations and findings presented in the existing literature focused on best practices with respect to workflow design, implementation and communication of workflow-centric research. These recommendations were summarised and characterised to identify the fundamental factors that should be considered carefully for complete provenance documentation and its application. This contribution was presented in *Chapter 4* meets our objective: *"To extend the understanding of such resources from O3 by revisiting literature dedicated to designing best practise recommendations for workflow-centric research and compiling these recommendations"* **(**O4).

To answer the research question **RQ3**: ***"How can we devise a hierarchical provenance framework encompassing community experiences that can serve as a guiding principle to determine the state of provenance of a given published***

analysis designed using a particular workflow definition approach?", we have made the following contribution.

C5: Conceptual Provenance Framework

This research made a (theoretical) contribution by modelling a hierarchical conceptual provenance framework. Given the considerable heterogeneous nature of workflow-centric studies due to the different workflow definition approaches, implementations and choice of sharing mechanisms, devising a rigid framework dependent on a specific technical environment is unlikely to address the requirements of all workflow studies. Therefore a conceptual incremental framework was proposed that can be followed irrespective of the computing platforms and specific computational methods used, to analyse and maintain a certain level of provenance for workflow-centric analyses. The explicit enumeration of the levels of this framework help quantify the effort required to re-use a workflow and reproduce a given experiment. This contribution presented in *Chapter 4* meets our objective: *"To devise a conceptual provenance framework utilising the compilation done in O4 to capture the fundamental artefacts identified in O3 and O4 in a hierarchical fashion"* **(**O5).

To answer the research question **RQ4**: ***"How can we leverage existing abstraction and standardisation techniques to realise the provenance framework and demonstrate its utility?"***, we have made following contributions.

C6: *CWLProv* –Standard Format

This contribution actualises the hierarchical provenance framework by identifying and choosing the workflow definition, provenance representation and resource aggregation methods that support interoperable and comprehensive workflow-centric research. Leveraging initiatives such as CWL, re-

search objects and PROV-DM, we devised *CWLProv* –a format that structures the artefacts associated with any CWL workflow enactment. The choice of workflow definition approach is being widely adopted by various platforms and extends the impact of this contribution to all CWL-enabled platforms. *CWLProv* research objects provide a means for consistent inter-WMS and inter-computing platform communication of the research process due to their standardised nature and well-defined serialisation format. In this case we utilised BagIt which makes validation possible through use of existing BagIt libraries. The contribution presented in *Chapter 5* meets our objective: *"To demonstrate the working of the framework from O5 by formulating a standardised format for interoperable workflow-centric analysis representation"* **(**O6).

C7: Practical Implementation

We have extended an existing workflow executor –*cwltool* for CWL workflows, by implementing *CWLProv* and demonstrating the utility of the provenance framework (**C5**) and how it leverages standardised representation (**C6**). This implementation is available as an optional module which when invoked creates annotated and provenance-rich workflow-centric research objects by aggregating and preserving data and methods utilised in a given workflow enactment. The evaluation of **C5** and **C6** was made possible with this implementation. This contribution presented in *Chapter 5* partially meets our objective: *"To assess the effectiveness of the solution proposed in O5 and O6 with real-world bioinformatics workflows."* **(**O7).

C8: Interoperability Demonstration

A case-based evaluation of **C5** and **C6** utilising **C7** to enact three commonly implemented real-life bioinformatics workflows designed by independent

groups provided a clear demonstration of interoperability of the *CWLProv* ROs across different (heterogeneous) platforms. This contribution presented in *Chapter 6* partially meets our objective: *"To assess the effectiveness of the solution proposed in O5 and O6 with real-world bioinformatics workflows."* **(**O7).

C9: Supporting Tool Development

We have contributed to the existing CWL-related tool ecosystem[5] by developing *cwlprov-py*, a command line tool and Python API to explore *CWLProv* ROs and interpret the captured provenance of CWL workflows. This tool is currently supporting several use-cases including enabling partial re-runs of a specific sub-workflow/workflow step by extracting requirements from the provenance profile and utilising the aggregated resources in the respective *CWLProv* RO. This contribution presented in *Chapter 6* partially meets our objective: *"To assess the effectiveness of the solution achieved after meeting O5 and O6 with real-world bioinformatics workflows."* **(**O7).

7.2 Limitations & Future Work

In this section we discuss some methodological and practical limitations of the work undertaken in this thesis and identify potential future research directions.

The empirical case study presented in *Chapter 3* can be extended in two ways.

1. First, the work is linked to using a single exemplar system from each workflow definition approach. Future work could focus on extending the case study to include other WMSs such as Taverna, bcbio-nextgen, nextflow or

[5]https://www.commonwl.org/

BioWorkbench and then re-evaluating the implicit assumptions made in the different approaches and their impact on provenance.

2. Galaxy was used in the case study in *Chapter 3*. However, since the time of conducting the case study, Galaxy has introduced various new features relevant to provenance, including support for container-based technologies and Bioconda recipes; enhancements to the Galaxy platform include conda auto initialisation, hash tags for data/collections, Singularity support and options for choosing specific archived tool versions. These features should be included in future studies to determine the impact on workflow provenance and handling of implicit assumptions.

The practical implementation of *CWLProv* was only done for only one workflow executor, *cwltool*. Extension of WMSs supporting CWL to enable support for effective sharing of workflow artefacts is another direction for future work. The evaluation activity included only two executors where the *CWLProv* RO was generated by only one executor (*cwltool*). With more implementations in place, this evaluation activity could be extended across different WMSs.

As the standards and approaches leveraged in this research are domain-agnostic, the findings and principles can in principle be applied to any scientific workflow irrespective of the research domain. However, as this thesis focuses on the provenance challenges of bioinformatics workflows, the requirement analysis and testing has been carried out using bioinformatics workflows. The requirement analysis activities such as the empirical study and literature exploration could be carried out in other domains to explore, enhance and contextualise the provenance framework and determine the extent that it meets the requirements specific to the given research area.

In addition to the above, several directions are discussed in Section 6.5 that

could be explored. An important limitation of the open access and sharing mechanism advocated strongly in this thesis is data security and privacy, which typifies the *-omics*, biomedical and clinical domains. This limitation can be a practical bottleneck in adopting the proposed format for sharing content addressable data artefacts. This will adversely impact the open publication and transparent sharing of artefacts utilised in a workflow analysis. We have not explored data privacy in this thesis however it would be a tremendous benefit if future research can contribute in this direction.

7.3 Impact

The contributions of this thesis have already impacted recent work by other researchers. The empirical case study and its findings are already contributing in shaping recent research [213] as stated below:

"They show a set of aspects associated with each SWfMS that hinder the understanding and reproducibility of workflows, including lack of documentation and provenance data. For each aspect, the authors propose recommendations that, together with provenance patterns, could facilitate reproducibility. Our approach supports some recommendations raised by the work, such as the availability of workflows and the framework through public repositories. In addition, through machine learning techniques, we demonstrate that it is also possible to predict the execution time of workflows based on provenance data from previous executions. In future work, the use of these techniques may support other recommendations raised by the authors."

The conceptual hierarchical provenance framework and *CWLProv* format are now utilised as a guide by the Nextflow team to implement research object support for the nextflow pipelines [219, 220]. This is a clear representation of a

practical case where the provenance framework devised in this thesis has been employed by other independent groups for understanding the requirements of provenance without explicit dependence on a particular type of workflow specification. All of the WMSs supporting nextflow can benefit from the interoperable ROs that are generated through this work.

As a practical implementation for creating *CWLProv* RO, the *cwltool* can be used as a feature complete reference executor. Therefore, anyone using *cwltool* for CWL workflow enactment will directly benefit from this contribution. Indeed a recent effort focused on extending *toil-cwl-runner*[6] has commenced which will provide support for *CWLProv* RO generation after workflow enactment.

Overall this thesis has significant implications for the conceptual understanding of bioinformatics workflow design, resource sharing mechanisms, provenance capture and subsequent reuse. CWL, the workflow definition approach selected to implement *CWLProv*, has a rapidly growing audience with various active implementations enabling its support on different WMSs. As the work done in this thesis was not produced independently but rather through international collaborations with continuous feedback from the scientific community through formal meetings, e.g. during the Bioinformatics Open Source Conference (BOSC) 2017 and 2018, *CWLProv* can be considered as the core current approach for exchanging and sharing results of any CWL-based workflow enactment across all WMSs supporting CWL or indeed using it as a default workflow language.

WMSs without CWL support can also benefit from the contributions of this thesis. These systems can translate their native output objects and local provenance logs (if any) into *CWLProv* ROs and share these objects through open access repositories to make the outputs of the workflow enactments accessible and interoperable.

[6] https://github.com/DataBiosphere/toil/issues/2390

In summary, this thesis has presented an in depth analysis of the related literature, carried out practical and pragmatic investigations and made significant contributions towards improving the provenance capture and transparency of workflow-centric analyses. Overall the practical implementation and evaluation of the proposed solutions with respect to reproducibility and interoperability address the overarching hypothesis set in *Chapter 1*, namely:

Declarative and platform-agnostic workflow definitions with high degrees of abstraction that leverage standardised and provenance-aware resource aggregation can help address the reproducibility and interoperability issues present in workflow-centric scientific research.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Jill P Mesirov. "Accessible reproducible research". In: *Science* 327.5964 (2010), pp. 415–416.

[2] Ian J. Taylor et al. *Workflows for e-Science: Scientific Workflows for Grids*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 1846285194.

[3] Víctor Cuevas-Vicenttín et al. "Scientific workflows and provenance: Introduction and research opportunities". In: *Datenbank-Spektrum* 12.3 (2012), pp. 193–203.

[4] Yolanda Gil et al. "Examining the Challenges of Scientific Workflows". In: *Computer* 40.12 (Dec. 2007), pp. 24–32. DOI: 10.1109/mc.2007.421.

[5] Katherine Wolstencroft et al. "The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud." In: *Nucleic Acids Res* 41.Web Server issue (July 2013), W557–61. DOI: 10.1093/nar/gkt328.

[6] Juliana Freire. "Making computations and publications reproducible with vistrails". In: *Computing in Science & Engineering* 14.4 (2012), pp. 18–25.

[7] Shawn Bowers et al. "Kepler/pPOD: Scientific Workflow and Provenance Support for Assembling the Tree of Life". In: *Provenance and Annotation of Data and Processes*. Ed. by Juliana Freire, David Koop, and Luc Moreau. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 70–77. ISBN: 978-3-540-89965-5. DOI: 10.1007/978-3-540-89965-5_9.

[8] Enis Afgan et al. "The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update". In: *Nucleic acids research* 46.W1 (2018), W537–W544.

[9] Heidi Kuehn et al. "Using GenePattern for Gene Expression Analysis". In: *Current Protocols in Bioinformatics* 22.1 (June 2008), pp. 7.12.1–7.12.39. DOI: 10.1002/0471250953.bi0712s22.

[10] Nayanah Siva. "1000 Genomes project". In: *Nature biotechnology* 26.3 (2008), pp. 256–256.

[11] P Rice, I Longden, and A Bleasby. "EMBOSS: The European molecular biology open software suite". In: *Trends in genetics* 16.6 (2000), pp. 276–7.

[12] Jason E Stajich et al. "The Bioperl toolkit: Perl modules for the life sciences". In: *Genome research* 12.10 (2002), pp. 1611–1618.

[13] Peter JA Cock et al. "Biopython: freely available Python tools for computational molecular biology and bioinformatics". In: *Bioinformatics* 25.11 (2009), pp. 1422–1423.

[14] David F Ransohoff. "Promises and limitations of biomarkers". In: *Cancer prevention II*. Springer, 2009, pp. 55–59.

[15] Christine M Micheel, Sharly J Nass, Gilbert S Omenn, et al. *Evolution of translational omics: lessons learned and the path forward*. National Academies Press, 2012.

[16] Christina L. Zheng et al. "Use of semantic workflows to enhance transparency and reproducibility in clinical omics". In: *Genome Medicine* 7.1 (July 2015). DOI: 10.1186/s13073-015-0202-y.

[17] Lisa M McShane et al. "Criteria for the use of omics-based predictors in clinical trials". In: *Nature* 502.7471 (2013), p. 317.

[18] "Genome in a bottle—a human DNA standard". In: *Nature Biotechnology* 33.7 (July 2015), pp. 675–675. DOI: 10.1038/nbt0715-675a.

[19] Anton Nekrutenko and James Taylor. "Next-generation sequencing data interpretation: enhancing reproducibility and accessibility". In: *Nature reviews genetics* 13.9 (2012), pp. 667–672.

[20] Keith A Baggerly and Kevin R Coombes. *What information should be required to support clinical "omics" publications?* 2011.

[21] David De Roure et al. "Towards the preservation of scientific workflows". In: *Procs. of the 8th International Conference on Preservation of Digital Objects (iPRES 2011). ACM*. 2011.

[22] Jun Zhao et al. "Why workflows break –Understanding and combating decay in Taverna workflows". In: *IEEE 8th international conference on e-Science*. IEEE. 2012, pp. 1–9.

[23] Nicolas Stransky et al. "The mutational landscape of head and neck squamous cell carcinoma". In: *Science* 333.6046 (2011), pp. 1157–1160.

[24] Luc Moreau et al. "The Open Provenance Model: An Overview". In: *Provenance and Annotation of Data and Processes*. Ed. by Juliana Freire, David Koop, and Luc Moreau. Springer Berlin Heidelberg, 2008, pp. 323–326. ISBN: 978-3-540-89965-5.

[25] *PROV-Overview. An Overview of the PROV Family of Documents*. Project Report. Apr. 2013. URL: https://eprints.soton.ac.uk/356854/.

[26] Carl Lagoze et al. "Object Re-Use Exchange: A Resource-Centric Approach". In: *CoRR* abs/0804.2273 (2008).

[27] William Michener et al. "DataONE: Data Observation Network for Earth - Preserving Data and Enabling Innovation in the Biological and Environmental Sciences". In: *D-Lib Magazine* 17.1/2 (Jan. 2011). DOI: 10.1045/january2011-michener.

[28] Fernando Chirigati et al. "Reprozip: Computational reproducibility with ease". In: *Proceedings of the 2016 International Conference on Management of Data*. ACM. 2016, pp. 2085–2088.

[29] Khalid Belhajjame et al. "Using a suite of ontologies for preserving workflow-centric research objects". In: *Journal of Web Semantics* 32 (May 2015), pp. 16–42. DOI: 10.1016/j.websem.2015.01.003.

[30] Paolo Missier. "The Lifecycle of Provenance Metadata and Its Associated Challenges and Opportunities". In: *Building Trust in Information*. Springer International Publishing, 2016, pp. 127–137. DOI: 10.1007/978-3-319-40226-0_8.

[31] Peter Amstutz et al. *Common Workflow Language, v1.0*. 2016. DOI: 10.6084/m9.figshare.3115156.v2.

[32] Khalid Belhajjame et al. *PROV-DM: The PROV Data Model*. Project Report. Apr. 2013. URL: https://eprints.soton.ac.uk/356851/.

[33] Paolo Missier, Khalid Belhajjame, and James Cheney. "The W3C PROV family of specifications for modelling provenance metadata". In: *Proceedings of the 16th international conference on extending database technology*. ACM. 2013, pp. 773–776.

[34] Ewa Deelman et al. "Workflows and e-Science: An overview of workflow system features and capabilities". In: *Future generation computer systems* 25.5 (2009), pp. 528–540.

[35] Carole Goble. *Being FAIR: Enabling Reproducible Data Science*. Accessed October 2018. 2018. URL: https://www.slideshare.net/carolegoble/being-fair-enabling-reproducible-data-science.

[36] Rajendra Bose and James Frew. "Lineage retrieval for scientific data processing: a survey". In: *ACM Computing Surveys* 37.1 (Mar. 2005), pp. 1–28. DOI: 10.1145/1057977.1057978.

[37] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. "Data Provenance: Some Basic Issues". In: *FST TCS 2000: Foundations of Software Technology and Theoretical Computer Science*. Ed. by Sanjiv Kapoor and Sanjiva Prasad. Springer Berlin Heidelberg, 2000, pp. 87–93.

[38] Gustavo Alonso and Claus Hagen. "Geo-Opera: Workflow concepts for spatial processes". In: *Advances in Spatial Databases*. Ed. by Michel Scholl and Agnès Voisard. Springer Berlin Heidelberg, 1997, pp. 238–258. ISBN: 978-3-540-69240-9.

[39] Victor Hock Kim Tan. "Interaction tracing for mobile agent security". PhD thesis. Engineering and Applied Science, 2004.

[40] Pinar Alper et al. "Small Is Beautiful: Summarizing Scientific Workflows Using Semantic Annotations". In: *2013 IEEE International Congress on Big Data*. IEEE, June 2013. DOI: 10.1109/bigdata.congress.2013.49.

[41] T. Bench-Capon et al. "Two aspects of the validation and verification of knowledge-based systems". In: *IEEE Expert* 8.3 (June 1993), pp. 76–81. DOI: 10.1109/64.215226.

[42] David Tennenhouse. "Proactive computing". In: *Communications of the ACM* 43.5 (May 2000), pp. 43–50. DOI: 10.1145/332833.332837.

[43] James Cheney et al. "Provenance: a future history". In: *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*. ACM. 2009, pp. 957–964.

[44] Luc Moreau et al. "The foundations for provenance on the web". In: *Foundations and Trends® in Web Science* 2.2–3 (2010), pp. 99–241. DOI: 10.1561/1800000010.

[45] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. "A survey of data provenance in e-science". In: *ACM SIGMOD Record* 34.3 (Sept. 2005), p. 31. DOI: 10.1145/1084805.1084812.

[46] Jun Zhao et al. "Annotating, linking and browsing provenance logs for e-Science". In: *In Proc. of the Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data*. 2003, pp. 158–176.

[47] Paul Groth et al. "An architecture for provenance systems". In: (2006).

[48] Peter Buneman et al. "Archiving scientific data". In: *ACM Transactions on Database Systems* 29.1 (Mar. 2004), pp. 2–42. DOI: 10.1145/974750.974752.

[49] Simon Miles et al. "The requirements of using provenance in e-science experiments". In: 5.1 (Mar. 2007), pp. 1–25. ISSN: 1572-9184. DOI: 10.1007/s10723-006-9055-3.

[50] Peter Buneman, Sanjeev Khanna, and Tan Wang-Chiew. "Why and where: A characterization of data provenance". In: *Database Theory — ICDT 2001*. Springer Berlin Heidelberg, 2001, pp. 316–330. ISBN: 978-3-540-44503-6.

[51] Todd J Green, Grigoris Karvounarakis, and Val Tannen. "Provenance semirings". In: *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM. 2007, pp. 31–40.

[52] Yingwei Cui and Jennifer Widom. "Practical lineage tracing in data warehouses". In: *Data Engineering, 2000. Proceedings. 16th International Conference on*. IEEE. 2000, pp. 367–378.

[53] Bertram Ludäscher. "A brief tour through provenance in scientific workflows and databases". In: *Building Trust in Information*. Springer, 2016, pp. 103–126.

[54] Wang Chiew Tan et al. "Provenance in databases: past, current, and future." In: *IEEE Data Eng. Bull.* 30.4 (2007), pp. 3–12.

[55] James Cheney, Laura Chiticariu, Wang-Chiew Tan, et al. "Provenance in databases: Why, how, and where". In: *Foundations and Trends® in Databases* 1.4 (2009), pp. 379–474.

[56] Wang Chiew Tan. "Research problems in data provenance." In: *IEEE Data Eng. Bull.* 27.4 (2004), pp. 45–52.

[57] Boris Glavic and Klaus R Dittrich. "Data Provenance: A Categorization of Existing Approaches." In: *BTW*. Vol. 7. 12. 2007, pp. 227–241.

[58] David Hollingsworth and UK Hampshire. "Workflow management coalition: The workflow reference model". In: *Document Number TC00-1003* 19 (1995), p. 16.

[59] *Web services business process execution language version 2.0*. Accessed July 2018. 2007. URL: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.

[60] W.M.P. van der Aalst and A.H.M. ter Hofstede. "YAWL: yet another workflow language". In: *Information Systems* 30.4 (June 2005), pp. 245–275. DOI: 10.1016/j.is.2004.02.002.

[61] Nathaniel Palmer. "XML Process Definition Language". In: *Encyclopedia of Database Systems* (2016), pp. 1–1.

[62] Nickolas Kavantzas et al. "Web services choreography description language version 1.0". In: *W3C candidate recommendation* 9 (2005), pp. 290–313.

[63] Bertram Ludäscher et al. "Scientific workflow management and the Kepler system". In: *Concurrency and Computation: Practice and Experience* 18.10 (2006), pp. 1039–1065.

[64] Ilkay Altintas, Oscar Barney, and Efrat Jaeger-Frank. "Provenance Collection Support in the Kepler Scientific Workflow System". In: *Provenance and Annotation of Data*. Ed. by Luc Moreau and Ian Foster. Springer Berlin Heidelberg, 2006, pp. 118–132. ISBN: 978-3-540-46303-0.

[65] Shawn Bowers. *Scientific Workflow, Provenance, and Data Modeling Challenges and Approaches*. May 2012. DOI: 10.1007/s13740-012-0004-y.

[66] Bertram Ludäscher et al. "Scientific Workflows: Business as Usual?" In: *Business Process Management*. Ed. by Umeshwar Dayal et al. Springer Berlin Heidelberg, 2009, pp. 31–47. ISBN: 978-3-642-03848-8.

[67] Adam Barker and Jano van Hemert. "Scientific Workflow: A Survey and Research Directions". In: *Parallel Processing and Applied Mathematics*. Ed. by Roman Wyrzykowski et al. Springer Berlin Heidelberg, 2008, pp. 746–753. ISBN: 978-3-540-68111-3.

[68] Susan B Davidson et al. "Provenance in scientific workflow systems." In: *IEEE Data Eng. Bull.* 30.4 (2007), pp. 44–50.

[69] Pinar Alper et al. "Enhancing and Abstracting Scientific Workflow Provenance for Data Publishing". In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. EDBT '13. Genoa, Italy: ACM, 2013, pp. 313–318. ISBN: 978-1-4503-1599-9. DOI: 10.1145/2457317.2457370.

[70] Brian F Lavoie. "The open archival information system reference model: Introductory guide". In: *Microform & imaging review* 33.2 (2004), pp. 68–81.

[71] Tim Booher. *Provenance (in Computer Science)*. Accessed September 2018. Mar. 2015. URL: https://www.theboohers.org/2015/03/03/provenance/.

[72] Shawn Bowers and Bertram Ludäscher. "Actor-Oriented Design of Scientific Workflows". In: *Conceptual Modeling – ER 2005*. Ed. by Lois Delcambre et al. Springer Berlin Heidelberg, 2005, pp. 369–384. ISBN: 978-3-540-32068-5.

[73] Shirley Cohen, Sarah Cohen-Boulakia, and Susan Davidson. "Towards a Model of Provenance and User Views in Scientific Workflows". In: *Data Integration in the Life Sciences*. Ed. by Ulf Leser, Felix Naumann, and Barbara Eckman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 264–279. ISBN: 978-3-540-36595-2. DOI: 10.1007/11799511_24.

[74] Yong Zhao, Michael Wilde, and Ian Foster. "Applying the Virtual Data Provenance Model". In: *Provenance and Annotation of Data*. Ed. by Luc Moreau and Ian Foster. Springer Berlin Heidelberg, 2006, pp. 148–161. ISBN: 978-3-540-46303-0. DOI: 10.1007/11890850_16.

[75] Ben Clifford et al. "Tracking provenance in a virtual data grid". In: *Concurrency and Computation: Practice and Experience* 20.5 (2008), pp. 565–575. DOI: 10.1002/cpe.1256.

[76] Mark Robinson et al. "CWL Viewer: the common workflow language viewer". In: *F1000Research* 6.ISCB Comm J (July 2017). Accessed July 2018, 1075 (poster). URL: https://doi.org/10.7490/f1000research.1114375.1.

[77] David De Roure and Carole Goble. "Anchors in shifting sand: the primacy of method in the web of data". In: (2010).

[78] Chunhyeok Lim et al. “Prospective and retrospective provenance collection in scientific workflow environments”. In: *IEEE International Conference on Services computing*. IEEE. 2010, pp. 449–456. DOI: 10.1109/scc.2010.18.

[79] Paolo Missier et al. “Provenance and Data Differencing for Workflow Reproducibility Analysis”. In: *Concurr. Comput. : Pract. Exper.* 28.4 (Mar. 2016), pp. 995–1015. ISSN: 1532-0626. DOI: 10.1002/cpe.3035.

[80] Fabio Casati et al. “Workflow evolution”. In: *Data & Knowledge Engineering* 24.3 (1998), pp. 211–238.

[81] Juliana Freire, Philippe Bonnet, and Dennis Shasha. “Computational Reproducibility: State-of-the-art, Challenges, and Database Research Opportunities”. In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’12. Scottsdale, Arizona, USA: ACM, 2012, pp. 593–596. ISBN: 978-1-4503-1247-9. DOI: 10.1145/2213836.2213908.

[82] Eran Chinthaka Withana et al. “Versioning for workflow evolution”. In: *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. ACM. 2010, pp. 756–765. DOI: 10.1145/1851476.1851586.

[83] Carole Goble. “Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics”. In: *Workshop on Data Derivation and Provenance, Chicago*. Vol. 3. 2002.

[84] Paul Groth et al. “Requirements for Provenance on the Web”. In: *International Journal of Digital Curation* 7.1 (Mar. 2012), pp. 39–56. DOI: 10.2218/ijdc.v7i1.213.

[85] Eric D Ragan et al. “Characterizing provenance in visualization and data analysis: an organizational framework of provenance types and purposes”. In: *IEEE transactions on visualization and computer graphics* 22.1 (2016), pp. 31–40. ISSN: 1077-2626. DOI: 10.1109/TVCG.2015.2467551.

[86] Pinar Alper et al. “LabelFlow: Exploiting Workflow Provenance to Surface Scientific Data Provenance”. In: *Provenance and Annotation of Data and Processes*. Ed. by Bertram Ludäscher and Beth Plale. Cham: Springer International Publishing, 2015, pp. 84–96. ISBN: 978-3-319-16462-5. DOI: 10.1007/978-3-319-16462-5_7.

[87] Daniel Garijo, Oscar Corcho, and Yolanda Gil. “Detecting Common Scientific Workflow Fragments Using Templates and Execution Provenance”. In: *Proceedings of the Seventh International Conference on Knowledge Capture*. K-CAP ’13. Banff, Canada: ACM, 2013, pp. 33–40. ISBN: 978-1-4503-2102-0. DOI: 10.1145/2479832.2479848.

[88] Eleanor Ainy et al. "Approximated summarization of data provenance". In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM. 2015, pp. 483–492. DOI: 10.1145/2806416.2806429.

[89] Daniel Garijo et al. "Quantifying reproducibility in computational biology: the case of the tuberculosis drugome." In: *PLoS ONE* 8.11 (Nov. 2013), e80278. DOI: 10.1371/journal.pone.0080278.

[90] Fiona Murphy. *Decision Trees: Licenses, Attribution, Provenance, Credit and Glitches*. Accessed October 2018. Sept. 2017. URL: https://www.force11.org/blog/decision-trees-licenses-attribution-provenance-credit-and-glitches.

[91] Khalid Belhajjame et al. "Workflow-centric research objects: First class citizens in scholarly discourse". In: *Proceedings of the 2nd Workshop on Semantic Publishing (SePublica 2012)*. Vol. 903. CEUR Workshop Proceedings. Accessed August 2017. 2012, pp. 1–12. URL: http://ceur-ws.org/Vol-903/paper-01.pdf.

[92] Luiz M. Gadelha and Marta Mattoso. "Applying Provenance to Protect Attribution in Distributed Computational Scientific Experiments". In: *Revised Selected Papers of the 5th International Provenance and Annotation Workshop on Provenance and Annotation of Data and Processes - Volume 8628*. IPAW 2014. Cologne, Germany: Springer-Verlag New York, Inc., 2015, pp. 139–151. ISBN: 978-3-319-16461-8. DOI: 10.1007/978-3-319-16462-5_11.

[93] Simon Woodman et al. "Achieving Reproducibility by Combining Provenance with Service and Workflow Versioning". In: *Proceedings of the 6th Workshop on Workflows in Support of Large-scale Science*. WORKS '11. Seattle, Washington, USA: ACM, 2011, pp. 127–136. ISBN: 978-1-4503-1100-7. DOI: 10.1145/2110497.2110512.

[94] Roger D. Peng. "Reproducible Research in Computational Science". In: *Science* 334.6060 (2011), pp. 1226–1227. ISSN: 0036-8075. DOI: 10.1126/science.1213847. eprint: http://science.sciencemag.org/content/334/6060/1226.full.pdf.

[95] Anna Bánáti, Péter Kacsuk, and Miklós Kozlovszky. "Four level provenance support to achieve portable reproducibility of scientific workflows". In: *Information and communication technology, electronics and microelectronics (MIPRO)*. IEEE. 2015, pp. 241–244. DOI: 10.1109/mipro.2015.7160272.

[96] Simon Woodman, Hugo Hiden, and Paul Watson. "Applications of provenance in performance prediction and data storage optimisation". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 299–309. DOI: 10.1016/j.future.2017.01.003.

[97] V. Curcin et al. "Implementing interoperable provenance in biomedical research". In: *Future Generation Computer Systems* 34 (May 2014), pp. 1–16. DOI: 10.1016/j.future.2013.12.001.

[98] Donovan Artz and Yolanda Gil. "A survey of trust in computer science and the Semantic Web". In: *Journal of Web Semantics* 5.2 (June 2007), pp. 58–71. DOI: 10.1016/j.websem.2007.03.002.

[99] Matthew Gamble and Carole Goble. "Quality, Trust, and Utility of Scientific Data on the Web: Towards a Joint Model". In: *Proceedings of the 3rd International Web Science Conference*. WebSci '11. Koblenz, Germany: ACM, 2011, 15:1–15:8. ISBN: 978-1-4503-0855-7. DOI: 10.1145/2527031.2527048.

[100] Michael Factor et al. "Authenticity and Provenance in Long Term Digital Preservation: Modeling and Implementation in Preservation Aware Storage". In: *First Workshop on on Theory and Practice of Provenance*. TAPP'09. San Francisco, CA: USENIX Association, 2009, 6:1–6:10. URL: http://dl.acm.org/citation.cfm?id=1525932.1525938.

[101] Daniel de Oliveira et al. "Debugging Scientific Workflows with Provenance: Achievements and Lessons Learned." In: *SBBD*. 2014, pp. 67–76.

[102] Luc Moreau et al. "Special Issue: The First Provenance Challenge". In: *Concurr. Comput. : Pract. Exper.* 20.5 (Apr. 2008), pp. 409–418. ISSN: 1532-0626. DOI: 10.1002/cpe.v20:5.

[103] Luc Moreau et al. "The Open Provenance Model core specification (v1.1)". In: *Future Generation Computer Systems* 27.6 (2011), pp. 743–756. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2010.07.005.

[104] *W3C Provenance Incubator Group Wiki - XG Provenance Wiki*. Accessed September 2018. URL: https://www.w3.org/2005/Incubator/prov/wiki/W3C_Provenance_Incubator_Group_Wiki.

[105] Yolanda Gil et al. "Provenance xg final report". In: *W3C Incubator Group Reports* (2010). URL: https://www.w3.org/2005/Incubator/prov/XGR-prov-20101214/.

[106] Stuart Weibel. "The Dublin Core: a simple content description model for electronic resources". In: *Bulletin of the American Society for Information Science and Technology* 24.1 (1997), pp. 9–11. DOI: 10.1002/bult.70.

[107] Deborah L McGuinness, Frank Van Harmelen, et al. "OWL web ontology language overview". In: *W3C recommendation* 10.10 (2004), p. 2004.

[108] Satya S Sahoo and Amit P Sheth. *Provenir ontology: Towards a framework for escience provenance management*. 2009. URL: https://corescholar.libraries.wright.edu/knoesis/80.

[109] Paulo Pinheiro Da Silva et al. "Inference Web in Action: Lightweight Use of the Proof Markup Language". In: *Proceedings of the 7th International Conference on The Semantic Web*. ISWC '08. Karlsruhe, Germany: Springer-Verlag, 2008, pp. 847–860. ISBN: 978-3-540-88563-4. DOI: 10.1007/978-3-540-88564-1_55.

[110] Keith Alexander et al. "Describing Linked Datasets - On the Design and Usage of voiD, the 'Vocabulary of Interlinked Datasets". In: *LDOW*. 2009. URL: http://hdl.handle.net/10379/543.

[111] Victor Cuevas-Vicenttin et al. "Modeling and querying scientific workflow provenance in the D-OPM". In: *2012 SC Companion: High-Performance Computing, Networking, Storage and Analysis (SCC)*. IEEE. 2012, pp. 119–128. DOI: 10.1109/SC.Companion.2012.27.

[112] Daniel Garijo and Yolanda Gil. "A new approach for publishing workflows: abstractions, standards, and linked data". In: *Proceedings of the 6th workshop on Workflows in support of large-scale science*. ACM. 2011, pp. 47–56.

[113] Yolanda Gil et al. "Wings: Intelligent workflow-based design of computational experiments". In: *IEEE intelligent systems* 26.1 (2011), pp. 62–72. ISSN: 1541-1672. DOI: 10.1109/MIS.2010.9.

[114] *PROV Model Primer*. Accessed September 2018. 2013. URL: https://www.w3.org/TR/2013/NOTE-prov-primer-20130430/.

[115] Guillem Closa et al. "W3C PROV to describe provenance at the dataset, feature and attribute levels in a distributed environment". In: *Computers, Environment and Urban Systems* 64 (July 2017), pp. 103–117. DOI: 10.1016/j.compenvurbsys.2017.01.008.

[116] Daniel Garijo and Yolanda Gil. "Augmenting PROV with Plans in P-PLAN: Scientific Processes as Linked Data". In: *Proceedings of the 2nd International Workshop on Linked Science*. 2012.

[117] Paolo Missier et al. "D-PROV: Extending the PROV Provenance Model with Workflow Structure". In: *Proceedings of the 5th USENIX Workshop on the Theory and Practice of Provenance*. TaPP '13. Lombard, Illinois: USENIX Association, 2013, 9:1–9:7. URL: http://dl.acm.org/citation.cfm?id=2482949.2482961.

[118] Víctor Cuevas-Vicenttín et al. *Provone: A prov extension data model for scientific workflow provenance*. 2015.

[119] Flavio Costa et al. "Capturing and Querying Workflow Runtime Provenance with PROV: A Practical Approach". In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. EDBT '13. Genoa, Italy: ACM, 2013, pp. 282–289. ISBN: 978-1-4503-1599-9. DOI: 10.1145/2457317.2457365.

[120] Marta Mattoso et al. “Towards supporting the life cycle of large scale scientific experiments”. In: *International Journal of Business Process Integration and Management* 5.1 (2010), pp. 79–92. DOI: 10.1504/ijbpim.2010.033176.

[121] Elaine Angelino, Daniel Yamins, and Margo Seltzer. “StarFlow: A Script-Centric Data Analysis Environment”. In: *Provenance and Annotation of Data and Processes*. Ed. by Deborah L. McGuinness, James R. Michaelis, and Luc Moreau. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 236–250. ISBN: 978-3-642-17819-1. DOI: 10.1007/978-3-642-17819-1_27.

[122] Leonardo Murta et al. “noWorkflow: Capturing and Analyzing Provenance of Scripts”. In: *Provenance and Annotation of Data and Processes*. Ed. by Bertram Ludäscher and Beth Plale. Cham: Springer International Publishing, 2015, pp. 71–83. ISBN: 978-3-319-16462-5. DOI: 10.1007/978-3-319-16462-5_6.

[123] Timothy McPhillips et al. “YesWorkflow: A User-Oriented, Language Independent Tool for Recovering Workflow Information from Scripts”. In: *International Journal of Digital Curation* 10 (2015), 298–313.

[124] Saumen Dey et al. “Linking Prospective and Retrospective Provenance in Scripts”. In: (2015). URL: https://www.usenix.org/conference/tapp15/workshop-program/presentation/dey.

[125] João Felipe Pimentel et al. “Yin & Yang: Demonstrating Complementary Provenance from noWorkflow & YesWorkflow”. In: *Provenance and Annotation of Data and Processes*. Ed. by Marta Mattoso and Boris Glavic. Cham: Springer International Publishing, 2016, pp. 161–165. ISBN: 978-3-319-40593-3. DOI: 10.1007/978-3-319-40593-3_13.

[126] Domenico Talia. “Workflow Systems for Science: Concepts and Tools”. In: *ISRN Software Engineering* 2013 (2013), pp. 1–15. DOI: 10.1155/2013/404525.

[127] Malcolm Atkinson et al. “Scientific workflows: Past, present and future”. In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 216–227. DOI: 10.1016/j.future.2017.05.041.

[128] Timothy Lebo et al. “Prov-o: The prov ontology”. In: *W3C recommendation* 30 (2013).

[129] James Frew, Dominic Metzger, and Peter Slaughter. “Automatic capture and reconstruction of computational provenance”. In: *Concurrency and Computation: Practice and Experience* 20.5 (2008), pp. 485–496. DOI: 10.1002/cpe.1247.

[130] Adam Bates et al. “Trustworthy Whole-system Provenance for the Linux Kernel”. In: *Proceedings of the 24th USENIX Conference on Security Symposium*. SEC’15. Washington, D.C.: USENIX Association, 2015, pp. 319–334. ISBN: 978-1-931971-232. URL: http://dl.acm.org/citation.cfm?id=2831143.2831164.

[131] Shiqing Ma, Xiangyu Zhang, and Dongyan Xu. “ProTracer: Towards Practical Provenance Tracing by Alternating Between Logging and Tainting”. In: *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*. San Diego, CA, 2016.

[132] Manolis Stamatogiannakis, Paul Groth, and Herbert Bos. “Looking Inside the Black-Box: Capturing Data Provenance Using Dynamic Instrumentation”. In: *Provenance and Annotation of Data and Processes*. Ed. by Bertram Ludäscher and Beth Plale. Cham: Springer International Publishing, 2015, pp. 155–167. ISBN: 978-3-319-16462-5. DOI: 10.1007/978-3-319-16462-5_12.

[133] Dawood Tariq, Maisem Ali, and Ashish Gehani. “Towards Automated Collection of Application-level Data Provenance”. In: *Proceedings of the 4th USENIX Conference on Theory and Practice of Provenance*. TaPP’12. Boston, MA: USENIX Association, 2012, pp. 16–16. URL: http://dl.acm.org/citation.cfm?id=2342875.2342891.

[134] Vitor C Neves, Vanessa Braganholo, and Leonardo Murta. “Implicit provenance gathering through configuration management”. In: *Proceedings of the 5th International Workshop on Software Engineering for Computational Science and Engineering*. IEEE Press. 2013, pp. 92–95. DOI: 10.1109/SECSE.2013.6615105.

[135] Andrew P Davison. “Automated capture of experiment context for easier reproducibility in computational research”. In: *Computing in Science & Engineering* 14.4 (2012), pp. 48–56. DOI: 10.1109/MCSE.2012.41.

[136] Mohammad Rezwanul Huq, Peter M. G. Apers, and Andreas Wombacher. “ProvenanceCurious: A Tool to Infer Data Provenance from Scripts”. In: *Proceedings of the 16th International Conference on Extending Database Technology*. EDBT ’13. Genoa, Italy: ACM, 2013, pp. 765–768. ISBN: 978-1-4503-1597-5. DOI: 10.1145/2452376.2452475.

[137] Barbara S. Lerner and Emery R. Boose. “RDataTracker and DDG Explorer”. In: *Revised Selected Papers of the 5th International Provenance and Annotation Workshop on Provenance and Annotation of Data and Processes - Volume 8628*. IPAW 2014. Cologne, Germany: Springer-Verlag New York, Inc., 2015, pp. 288–290. ISBN: 978-3-319-16461-8. DOI: 10.1007/978-3-319-16462-5_36.

[138] Alban Gaignard, Khalid Belhajjame, and Hala Skaf-Molli. "SHARP: Harmonizing and Bridging Cross-Workflow Provenance". In: *The Semantic Web: ESWC 2017 Satellite Events*. Ed. by Eva Blomqvist et al. Cham: Springer International Publishing, 2017, pp. 219–234. ISBN: 978-3-319-70407-4. DOI: 10.1007/978-3-319-70407-4_35.

[139] Amir Sezavar Keshavarz, Trung Dong Huynh, and Luc Moreau. "Provenance for Online Decision Making". In: *Provenance and Annotation of Data and Processes*. Ed. by Bertram Ludäscher and Beth Plale. Cham: Springer International Publishing, 2015, pp. 44–55. ISBN: 978-3-319-16462-5. DOI: 10.1007/978-3-319-16462-5_4.

[140] Trung Dong Huynh et al. "Provenance Network Analytics". In: *Data Mining and Knowledge Discovery* 32.3 (May 2018), pp. 708–735. ISSN: 1573-756X. DOI: 10.1007/s10618-017-0549-3.

[141] Peng Chen, Beth Plale, and Mehmet S. Aktas. "Temporal representation for mining scientific data provenance". In: *Future Generation Computer Systems* 36 (July 2014), pp. 363–378. DOI: 10.1016/j.future.2013.09.032.

[142] Luc Moreau. "Aggregation by provenance types: A technique for summarising provenance graphs". In: *arXiv preprint arXiv:1504.02616* (2015). URL: https://arxiv.org/abs/1504.02616.

[143] Wellington Oliveira, Daniel De Oliveira, and Vanessa Braganholo. "Provenance Analytics for Workflow-Based Computational Experiments: A Survey". In: *ACM Comput. Surv.* 51.3 (May 2018), 53:1–53:25. ISSN: 0360-0300. DOI: 10.1145/3184900.

[144] Sarah Cohen-Boulakia et al. "Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 284–298. DOI: 10.1016/j.future.2017.01.012.

[145] Daniel G. Hurley, David M. Budden, and Edmund J. Crampin. "Virtual Reference Environments: a simple way to make research reproducible". In: *Briefings in Bioinformatics* 16.5 (Nov. 2014), pp. 901–903. DOI: 10.1093/bib/bbu043.

[146] Bill Howe. "Virtual Appliances, Cloud Computing, and Reproducible Research". In: *Computing in Science and Engg.* 14.4 (July 2012), pp. 36–41. ISSN: 1521-9615. DOI: 10.1109/MCSE.2012.62.

[147] Stephen Soltesz et al. "Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors". In: vol. 41. 3. New York, NY, USA: ACM, Mar. 2007, pp. 275–287. DOI: 10.1145/1272998.1273025.

[148] Joel T Dudley and Atul J Butte. "In silico research in the era of cloud computing". In: *Nature Biotechnology* 28.11 (Nov. 2010), pp. 1181–1185. DOI: 10.1038/nbt1110-1181.

[149] Jonathan Klinginsmith, Malika Mahoui, and Yuqing Melanie Wu. "Towards Reproducible eScience in the Cloud". In: *Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science*. CLOUDCOM '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 582–586. ISBN: 978-0-7695-4622-3. DOI: 10.1109/CloudCom.2011.89.

[150] Cristian Ruiz, Olivier Richard, and Joseph Emeras. "Reproducible Software Appliances for Experimentation". In: *Testbeds and Research Infrastructure: Development of Networks and Communities*. Ed. by Victor C.M. Leung et al. Cham: Springer International Publishing, 2014, pp. 33–42. ISBN: 978-3-319-13326-3. DOI: 10.1007/978-3-319-13326-3_4.

[151] *DigitalOcean - Cloud Computing, Simplicity at Scale*. Accessed December 2018. URL: https://www.digitalocean.com/.

[152] *Amazon EC2*. Accessed December 2018. URL: https://aws.amazon.com/ec2/.

[153] *Google Cloud including GCP & G Suite url = https://cloud.google.com/, note = Accessed December 2018*.

[154] *Microsoft Azure Cloud Computing Platform & Services*. Accessed December 2018. URL: https://azure.microsoft.com/en-us/.

[155] Boden Russell. *KVM and Docker LXC Benchmarking with OpenStack*. 2014. URL: http://bodenr.blogspot.com/2014/05/kvm-and-docker-lxc-benchmarking-with.html.

[156] Bjorn Gruening et al. "Recommendations for the packaging and containerizing of bioinformatics software". In: *F1000Research* 7 (2018). DOI: 10.12688/f1000research.15140.1.

[157] Brett K Beaulieu-Jones and Casey S Greene. "Reproducibility of computational workflows is automated using continuous analysis". In: *Nature Biotechnology* 35.4 (Mar. 2017), pp. 342–346. DOI: 10.1038/nbt.3780.

[158] Gregory M Kurtzer, Vanessa Sochat, and Michael W Bauer. "Singularity: Scientific containers for mobility of compute." In: *PLoS ONE* 12.5 (May 2017), e0177459. DOI: 10.1371/journal.pone.0177459.

[159] Björn Grüning et al. "Practical Computational Reproducibility in the Life Sciences". In: *Cell Systems* 6.6 (June 2018), pp. 631–635. DOI: 10.1016/j.cels.2018.03.014.

[160] Björn Grüning et al. "Bioconda: A sustainable and comprehensive software distribution for the life sciences". In: (Oct. 2017). DOI: 10.1101/207092.

[161] Cui Lin et al. "A Task Abstraction and Mapping Approach to the Shimming Problem in Scientific Workflows". In: *Proceedings of the 2009 IEEE International Conference on Services Computing*. SCC '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 284–291. ISBN: 978-0-7695-3811-2. DOI: 10.1109/SCC.2009.77.

[162] Susanna-Assunta Sansone et al. "Toward interoperable bioscience data". In: *Nature genetics* 44.2 (2012), p. 121. DOI: 10.1038/ng.1054.

[163] Frank T Bergmann et al. "COMBINE archive and OMEX format: one file to share all information to reproduce a modeling project". In: *BMC bioinformatics* 15.1 (2014), p. 369. DOI: 10.1186/s12859-014-0369-z.

[164] Vahan Simonyan, Jeremy Goecks, and Raja Mazumder. "Biocompute Objects—A Step towards Evaluation and Validation of Biomedical Scientific Computations". In: *PDA Journal of Pharmaceutical Science and Technology* 71.2 (Dec. 2016), pp. 136–146. DOI: 10.5731/pdajpst.2016.006734.

[165] Network Working Group. *draft-kunze-bagit-17 - The BagIt File Packaging Format (V1.0)*. Accessed September 2018. 2017. URL: https://tools.ietf.org/html/draft-kunze-bagit-17.

[166] Jose Manuel Gomez-Perez, Raul Palma, and Andres Garcia-Silva. "Towards a Human-Machine Scientific Partnership Based on Semantically Rich Research Objects". In: *2017 IEEE 13th International Conference on e-Science (e-Science)*. IEEE, Oct. 2017, pp. 266–275. DOI: 10.1109/eScience.2017.40.

[167] Kristina M Hettne et al. "Structuring research methods and data with the research object model: genomics workflows as a case study." In: *J Biomed Semantics* 5.1 (Sept. 2014), p. 41. DOI: 10.1186/2041-1480-5-41.

[168] Adnan Custovic et al. "The Study Team for Early Life Asthma Research (STELAR) consortium 'Asthma e-lab': team science bringing data, methods and investigators together". In: *Thorax* 70.8 (2015), pp. 799–801. ISSN: 0040-6376. DOI: 10.1136/thoraxjnl-2015-206781. eprint: https://thorax.bmj.com/content/70/8/799.full.pdf.

[169] Gil Alterovitz et al. *Enabling Precision Medicine via standard communication of NGS provenance, analysis, and results*. WorkingPaper. bioRxiv preprint. Accepted for PLOS Biology 5 Nov 2018. Sept. 2017, pp. 1–18. DOI: 10.1101/191783.

[170] Tam P Sneddon, Peter Li, and Scott C Edmunds. *GigaDB: announcing the GigaScience database*. July 2012. DOI: 10.1186/2047-217x-1-11.

[171] David De Roure, Carole Goble, and Robert Stevens. "Designing the myExperiment Virtual Research Environment for the Social Sharing of Workflows". In: E-SCIENCE '07 (2007), pp. 603–610. DOI: 10.1109/E-SCIENCE.2007.29.

[172] Merce Crosas. "The dataverse network®: an open-source application for sharing, discovering and preserving data". In: *D-lib Magazine* 17.1/2 (2011). URL: http://www.dlib.org/dlib/january11/crosas/01crosas.html.

[173] Juliana Freire et al. "Provenance for Computational Tasks: A Survey". In: *Computing in Science and Engg.* 10.3 (May 2008), pp. 11–21. ISSN: 1521-9615. DOI: 10.1109/MCSE.2008.79.

[174] Common Workflow Language project. *Existing Workflow Systems*. Accessed September 2017. 2018. URL: https://s.apache.org/existing-workflow-systems.

[175] Jeremy Leipzig. "A review of bioinformatic pipeline frameworks". In: *Briefings in Bioinformatics* (Mar. 2016), bbw020. DOI: 10.1093/bib/bbw020.

[176] M. Janetschek, R. Prodan, and S. Benedict. "A workflow runtime environment for manycore parallel architectures". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 330–347. DOI: 10.1016/j.future.2017.02.029.

[177] Joseph P. Macker and Ian Taylor. "Orchestration and analysis of decentralized workflows within heterogeneous networking infrastructures". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 388–401. DOI: 10.1016/j.future.2017.01.007.

[178] Zachary D. Stephens et al. "Big Data: Astronomical or Genomical?" In: *PLOS Biology* 13.7 (July 2015), e1002195. DOI: 10.1371/journal.pbio.1002195.

[179] Ivo Glynne Gut. "New sequencing technologies". In: *Clinical and Translational Oncology* 15.11 (July 2013), pp. 879–881. DOI: 10.1007/s12094-013-1073-6.

[180] Ola Spjuth et al. "Experiences with workflows for automating data-intensive bioinformatics". In: *Biology direct* 10.1 (2015), p. 43. DOI: 10.1186/s13062-015-0071-8.

[181] Simon P Sadedin et al. "Cpipe: a shared variant detection pipeline designed for diagnostic settings". In: *Genome medicine* 7.1 (2015), p. 1. DOI: 10.1186/s13073-015-0191-x.

[182] Roman Valls Guimera. "bcbio-nextgen: Automated, distributed next-gen sequencing pipeline". In: *EMBnet.journal* 17.B (Feb. 2012), p. 30. DOI: 10.14806/ej.17.b.286.

[183] Kathleen M. Fisch et al. "Omics Pipe: a community-based framework for reproducible multi-omics data analysis". In: *Bioinformatics* 31.11 (Jan. 2015), pp. 1724–1728. DOI: 10.1093/bioinformatics/btv061.

[184] Olga Golosova et al. “Unipro UGENE NGS pipelines and components for variant calling, RNA-seq and ChIP-seq data analyses”. In: *PeerJ* 2 (2014), e644. DOI: 10.7717/peerj.644.

[185] J. Koster and S. Rahmann. “Snakemake–a scalable bioinformatics workflow engine”. In: *Bioinformatics* 28.19 (Aug. 2012), pp. 2520–2522. DOI: 10.1093/bioinformatics/bts480.

[186] Simon P. Sadedin, Bernard Pope, and Alicia Oshlack. “Bpipe: a tool for running and managing bioinformatics pipelines”. In: *Bioinformatics* 28.11 (Apr. 2012), pp. 1525–1526. DOI: 10.1093/bioinformatics/bts167.

[187] Leo Goodstadt. “Ruffus: a lightweight Python library for computational pipelines”. In: *Bioinformatics* 26.21 (Sept. 2010), pp. 2778–2779. DOI: 10.1093/bioinformatics/btq524.

[188] Ilkay Altintas et al. “Kepler: an extensible system for design and execution of scientific workflows”. In: *Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004.* IEEE. June 2004, pp. 423–424. DOI: 10.1109/SSDM.2004.1311241.

[189] Ewa Deelman et al. “Pegasus, a workflow management system for science automation”. In: *Future generation computer systems* 46 (2015), pp. 17–35. DOI: 10.1016/j.future.2014.10.008.

[190] Wendy A Warr. “Scientific workflow systems: Pipeline pilot and KNIME”. In: *Journal of computer-aided molecular design* 26.7 (2012), pp. 801–804. DOI: 10.1007/s10822-012-9577-7.

[191] Ricky J Sethi and Yolanda Gil. “Reproducibility in computer vision: Towards open publication of image analysis experiments as semantic workflows”. In: *2016 IEEE 12th International Conference on e-Science (e-Science)*. IEEE. Oct. 2016, pp. 343–348. DOI: 10.1109/eScience.2016.7870918.

[192] Matheus Hauder et al. “Making data analysis expertise broadly accessible through workflows”. In: *Proceedings of the 6th workshop on workflows in support of large-scale science*. ACM Press, 2011, pp. 77–86. DOI: 10.1145/2110497.2110507.

[193] Zhili Zhao and Adrian Paschke. “A survey on semantic scientific workflow”. In: *Semantic web journal IOS press* (2012), pp. 1–5.

[194] K Voss, J Gentry, and G Van der Auwera. *Full-stack genomics pipelining with GATK4+ WDL+ Cromwell [version 1; not peer reviewed]*. 2017. DOI: https://doi.org/10.7490/f1000research.1114634.1.

[195] Fabian A. Buske et al. “NGSANE: a lightweight production informatics framework for high-throughput data analysis”. In: *Bioinformatics* 30.10 (Jan. 2014), pp. 1471–1472. DOI: 10.1093/bioinformatics/btu036.

[196] Jason Li et al. "Bioinformatics pipelines for targeted resequencing and whole-exome sequencing of human and mouse genomes: A virtual appliance approach for instant deployment". In: *PLOS ONE (Public Library of Science)* 9 (2014). DOI: 10.1371/journal.pone.0095217.

[197] Jochen Singer et al. "NGS-pipe: a flexible, easily extendable and highly configurable framework for NGS analysis". In: *Bioinformatics* 34.1 (Aug. 2017). Ed. by Bonnie Berger, pp. 107–108. DOI: 10.1093/bioinformatics/btx540.

[198] Jason L. Causey et al. "DNAp: A Pipeline for DNA-seq Data Analysis". In: *Scientific Reports* 8.1 (Dec. 2018). DOI: 10.1038/s41598-018-25022-6. URL: http://par.nsf.gov/biblio/10062406.

[199] Sehrish Kanwal et al. "Experiences in implementing large-scale biomedical workflows on the cloud: Challenges in transitioning to the clinical domain". In: *Health informatics society Australia (HISA)*. 2016.

[200] Oliver Hofmann et al. "Community Development of Validated Variant Calling Pipelines". In: (Nov. 2013). DOI: 10.6084/m9.figshare.840461.v1. URL: https://figshare.com/articles/Community_Development_of_Validated_Variant_Calling_Pipelines/840461.

[201] *Arvados*. http://www.arvados.org/. Accessed August 2017.

[202] Daniel Blankenberg et al. "Dissemination of scientific software with Galaxy ToolShed". In: *Genome biology* 15.2 (2014), p. 403. DOI: 10.1186/gb4161.

[203] Carole A Goble et al. "myExperiment: a repository and social network for the sharing of bioinformatics workflows". In: *Nucleic acids research* 38.suppl_2 (2010), W677–W682. DOI: 10.1093/nar/gkq429.

[204] J. Bhagat et al. "BioCatalogue: a universal catalogue of web services for the life sciences". In: *Nucleic Acids Research* 38.Web Server (May 2010), W689–W694. DOI: 10.1093/nar/gkq394.

[205] Mark Bieda. "Kepler for 'omics bioinformatics". In: *Procedia Computer Science* 9 (2012), pp. 1635–1638. DOI: 10.1016/j.procs.2012.04.180.

[206] Ilkay Altintas. "Distributed Workflow-driven Analysis of Large-scale Biological Data Using Biokepler". In: *Proceedings of the 2Nd International Workshop on Petascal Data Analytics: Challenges and Opportunities*. PDAC '11. Seattle, Washington, USA: ACM, 2011, pp. 41–42. ISBN: 978-1-4503-1130-4. DOI: 10.1145/2110205.2110215.

[207] Trung Dong Huynh et al. "The PROV-JSON serialization". In: (2013).

[208] Gideon Juve et al. "Characterizing and Profiling Scientific Workflows". In: *Future Gener. Comput. Syst.* 29.3 (Mar. 2013), pp. 682–692. ISSN: 0167-739X. DOI: 10.1016/j.future.2012.08.015.

[209] Jens-S Vöckler et al. “Kickstarting remote applications”. In: *2nd International Workshop on Grid Computing Environments*. 2006, pp. 1–8. URL: http://pegasus.isi.edu/publications/kickstart.pdf.

[210] *Enabling End-to-end Experiment Sharing and Reuse with Workflows via Jupyter Notebooks*. Oct. 2017. DOI: 10.6084/m9.figshare.5482474.v1. URL: https://figshare.com/articles/Enabling_End-to-end_Experiment_Sharing_and_Reuse_with_Workflows_via_Jupyter_Notebooks/5482474/1.

[211] Fernando Chirigati et al. “VisTrails Provenance Traces for Benchmarking”. In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. EDBT ’13. Genoa, Italy: ACM, 2013, pp. 323–324. ISBN: 978-1-4503-1599-9. DOI: 10.1145/2457317.2457373.

[212] Phillip Mates et al. “CrowdLabs: Social Analysis and Visualization for the Sciences”. In: *Scientific and Statistical Database Management*. Ed. by Judith Bayard Cushing, James French, and Shawn Bowers. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 555–564. ISBN: 978-3-642-22351-8. DOI: 10.1007/978-3-642-22351-8_38.

[213] Maria Luiza Mondelli et al. “BioWorkbench: a high-performance framework for managing and analyzing bioinformatics experiments”. In: *PeerJ* 6 (Aug. 2018), e5551. DOI: 10.7717/peerj.5551.

[214] Michael Wilde et al. “Swift: A language for distributed parallel scripting”. In: *Parallel Computing* 37.9 (2011), pp. 633–652. DOI: 10.1016/j.parco.2011.05.005.

[215] Mark Hall et al. “The WEKA Data Mining Software: An Update”. In: *SIGKDD Explor. Newsl.* 11.1 (Nov. 2009), pp. 10–18. ISSN: 1931-0145. DOI: 10.1145/1656274.1656278.

[216] Vladimir Korkhov et al. “SHIWA workflow interoperability solutions for neuroimaging data analysis”. In: *Stud Health Technol Inform* 175 (2012).

[217] John Vivian et al. “Toil enables reproducible, open source, big biomedical data analyses”. In: *Nature Biotechnology* 35.4 (Apr. 2017), pp. 314–316. DOI: 10.1038/nbt.3772.

[218] Paolo Di Tommaso et al. “Nextflow enables reproducible computational workflows”. In: *Nature Biotechnology* 35.4 (Apr. 2017), pp. 316–319. DOI: 10.1038/nbt.3820.

[219] Edgar Garriga Nogales, Paolo Di Tommaso, and Cedric Notredame. “Nextflow Integration For The Research Object Specification”. en. In: (2018). DOI: 10.5281/zenodo.1323831. URL: https://zenodo.org/record/1323831.

[220] Edgar Garriga Nogales, Paolo Di Tommaso, and Cedric Notredame. *Nextflow integration for the Research Object Specification*. Oct. 2018. DOI: 10.5281/zenodo.1472384. URL: https://figshare.com/articles/Nextflow_integration_for_the_Research_Object_Specification/7262030/1.

[221] A. McKenna et al. "The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data". In: *Genome Research* 20.9 (July 2010), pp. 1297–1303. DOI: 10.1101/gr.107524.110.

[222] Geraldine A Van der Auwera et al. "From FastQ data to high-confidence variant calls: the genome analysis toolkit best practices pipeline". In: *Current protocols in bioinformatics* (2013), pp. 11–10.

[223] *GATK Variant Calling Workflow using Galaxy, CWL and Cpipe*. Accessed February 2016. URL: https://github.com/FarahZKhan/GATK-CaseStudy.

[224] Justin Zook. *New high-confidence NA12878 genotypes integrating phased pedigree calls*. https://sites.stanford.edu/abms/content/new-high-confidence-na12878-genotypes-integrating-phased-pedigree-calls. 2014.

[225] J Zook. *Want to better understand the accuracy of your human genome sequencing*. Accessed October 2016. 2013. URL: http://www.nist.gov/mml/bbd/ppgenomeinabottle2.cfm.

[226] *GATK — What's in the resource bundle and how can I get it?* Accessed November 2015. URL: https://software.broadinstitute.org/gatk/documentation/article.php?id=1213.

[227] *Illumina Sequencing Download*. Accessed November 2015. URL: https://support.illumina.com/sequencing/downloads.html.

[228] *Nectar Cloud - Nectar*. Accessed September 2018. URL: https://nectar.org.au/research-cloud/.

[229] *The open source version of the Melbourne Genomics Health Alliance Exome Sequencing Pipeline*. Accessed November 2015. URL: https://github.com/MelbourneGenomics/Cpipe.

[230] Heng Li. "Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM". In: *arXiv preprint arXiv:1303.3997* (2013). URL: https://arxiv.org/abs/1303.3997.

[231] William McLaren et al. "The ensembl variant effect predictor". In: *Genome biology* 17.1 (2016), p. 122.

[232] Kai Wang, Mingyao Li, and Hakon Hakonarson. "ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data". In: *Nucleic acids research* 38.16 (July 2010), e164–e164. DOI: 10.1093/nar/gkq603.

[233] Enis Afgan et al. "Genomics Virtual Laboratory: A Practical Bioinformatics Workbench for the Cloud". In: *PLOS ONE* 10.10 (Oct. 2015). Ed. by Christophe Antoniewski, e0140829. DOI: 10.1371/journal.pone.0140829.

[234] Peter Amstutz et al. *common-workflow-language/cwltool*. GitHub, 2017. URL: https://github.com/common-workflow-language/cwltool/releases/tag/1.0.20170828135420 (Accessed Aug. 28, 2017).

[235] *Picard Tools - By Broad Institute*. Accessed October 2015. URL: http://broadinstitute.github.io/picard/.

[236] *fzkhan/picard-1.136-gatk-2.8 - Docker Hub*. Accessed November 2015. URL: https://hub.docker.com/r/fzkhan/picard-1.136-gatk-2.8/.

[237] Katherine Wolstencroft et al. "The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud". In: *Nucleic acids research* (2013), gkt328.

[238] Michael L. Metzker. "Sequencing technologies — the next generation". In: *Nature Reviews Genetics* 11.1 (Dec. 2009), pp. 31–46. DOI: 10.1038/nrg2626. URL: https://doi.org/10.1038/nrg2626.

[239] Sehrish Kanwal et al. "Challenges of large-scale biomedical workflows on the cloud–A case study on the need for reproducibility of results". In: *2015 IEEE 28th International Symposium on Computer-Based Medical Systems*. IEEE. 2015, pp. 220–225. DOI: 10.1109/CBMS.2015.28.

[240] Antony T. Vincent and Steve J. Charette. "Freedom in bioinformatics". In: *Frontiers in Genetics* 5 (July 2014). DOI: 10.3389/fgene.2014.00259.

[241] *The Center for Open Science*. Accessed February 2016. URL: https://cos.io/.

[242] Daniel Garijo, Yolanda Gil, and Oscar Corcho. "Abstract, link, publish, exploit: An end to end framework for workflow sharing". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 271–283. DOI: 10.1016/j.future.2017.01.008.

[243] Geir Kjetil Sandve et al. "Ten Simple Rules for Reproducible Computational Research". In: *PLoS Computational Biology* 9.10 (Oct. 2013). Ed. by Philip E. Bourne, e1003285. DOI: 10.1371/journal.pcbi.1003285.

[244] Aravind Mohan, Shiyong Lu, and Alexander Kotov. "Addressing the shimming problem in big data scientific workflows". In: *Services Computing (SCC), 2014 IEEE International Conference on*. IEEE. June 2014, pp. 347–354. DOI: 10.1109/SCC.2014.53.

[245] Sehrish Kanwal et al. "Investigating reproducibility and tracking provenance—A genomic workflow case study". In: *BMC bioinformatics* 18.1 (2017), p. 337. DOI: 10.1186/s12859-017-1747-0.

[246] Richard Littauer et al. "Trends in Use of Scientific Workflows: Insights from a Public Repository and Recommendations for Best Practice". In: *International Journal of Digital Curation* 7.2 (Oct. 2012), pp. 92–100. DOI: 10.2218/ijdc.v7i2.232.

[247] V. Stodden et al. "Enhancing reproducibility for computational methods". In: *Science* 354.6317 (Dec. 2016), pp. 1240–1241. DOI: 10.1126/science.aah6168.

[248] Victoria Stodden and Sheila Miguez. "Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research". In: *Journal of Open Research Software* 2.1 (July 2014). DOI: 10.5334/jors.ay.

[249] Kristina M Hettne et al. "Best Practices for Workflow Design: How to Prevent Workflow Decay". In: *SWAT4LS*. 2012.

[250] Melissa Gymrek and Yossi Farjoun. "Recommendations for open data science". In: *GigaScience* 5.1 (May 2016). DOI: 10.1186/s13742-016-0127-4.

[251] Sonja Holl et al. "On Specifying and Sharing Scientific Workflow Optimization Results Using Research Objects". In: *Proceedings of the 8th Workshop on Workflows in Support of Large-Scale Science*. WORKS '13. Denver, Colorado: ACM, 2013, pp. 28–37. ISBN: 978-1-4503-2502-8. DOI: 10.1145/2534248.2534251.

[252] D Hull et al. "Treating shimantic web syndrome with ontologies". In: *Proceedings of the First Advanced Knowledge Technologies Workshop on Semantic Web Services (AKT-SWS04) KMi*, ed. by J Domingue, L Cabral, and M Enrico. Vol. 122. CEUR Workshop Proceedings. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-122/paper1.pdf. CEUR-WS.org, 2004.

[253] Julie A McMurry et al. "Identifiers for the 21st century: How to design, provision, and reuse persistent identifiers to maximize utility and impact of life science data." In: *PLoS Biol* 15.6 (June 2017), e2001414. DOI: 10.1371/journal.pbio.2001414.

[254] Martin Klein et al. "Scholarly Context Not Found: One in Five Articles Suffers from Reference Rot". In: *PLoS ONE* 9.12 (Dec. 2014). Ed. by Judit Bar-Ilan, e115253. DOI: 10.1371/journal.pone.0115253.

[255] J. Ison et al. "EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats". In: *Bioinformatics* 29.10 (Mar. 2013), pp. 1325–1332. DOI: 10.1093/bioinformatics/btt113.

[256] James Malone et al. "The Software Ontology (SWO): a resource for reproducibility in biomedical data analysis, curation and digital preservation". In: *Journal of biomedical semantics* 5.1 (2014), p. 25. DOI: 10.1186/2041-1480-5-25.

[257] *RNASelector as a CWL workflow - Common Workflow Language Viewer*. Accessed October 2018. URL: https://bit.ly/2r0m7Kz.

[258] *Open Definition 2.1 - Open Definition - Defining Open in Open Data, Open Content and Open Knowledge*. Accessed October 2018. URL: http://opendefinition.org/od/2.1/en/.

[259] *Apache License, Version 2.0 — Open Source Initiative*. Accessed October 2018. URL: https://opensource.org/licenses/Apache-2.0.

[260] *The MIT License — Open Source Initiative*. Accessed October 2018. URL: https://opensource.org/licenses/MIT.

[261] Creative Commons. *About The Licenses - Creative Commons*. Accessed October 2018. 2015. URL: https://creativecommons.org/licenses/.

[262] Ulf Toelch and Dirk Ostwald. "Digital open science—Teaching digital tools for reproducible and transparent research". In: *PLOS Biology* 16.7 (July 2018), e2006022. DOI: 10.1371/journal.pbio.2006022.

[263] Antoon Goderis et al. "Seven Bottlenecks to Workflow Reuse and Repurposing". In: *The Semantic Web – ISWC 2005*. Ed. by Yolanda Gil et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 323–337. ISBN: 978-3-540-32082-1. DOI: 10.1007/11574620_25.

[264] M. Bubak et al. "Evaluation of Cloud Providers for VPH Applications". In: *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing* (May 2013). DOI: 10.1109/ccgrid.2013.54.

[265] Samuel V. Angiuoli et al. "Resources and Costs for Microbial Sequence Analysis Evaluated Using Virtual Machines and Cloud Computing". In: *PLoS ONE* 6.10 (Oct. 2011). Ed. by Sarah K.Editor Highlander, e26624. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0026624.

[266] Weiwei Chen and Ewa Deelman. "Partitioning and Scheduling Workflows Across Multiple Sites with Storage Constraints". In: *Proceedings of the 9th International Conference on Parallel Processing and Applied Mathematics - Volume Part II*. PPAM'11. Torun, Poland: Springer-Verlag, 2012, pp. 11–20. ISBN: 978-3-642-31499-5. DOI: 10.1007/978-3-642-31500-8_2. URL: http://dx.doi.org/10.1007/978-3-642-31500-8_2.

[267] *GATK — BP Doc #11165 — Data pre-processing for variant discovery*. Accessed September 2018. 2018. URL: https://software.broadinstitute.org/gatk/best-practices/workflow?id=11165.

[268] Maciej Malawski, Kamil Figiela, and Jarek Nabrzyski. "Cost minimization for computational applications on hybrid cloud infrastructures". In: *Future Generation Computer Systems* 29.7 (Sept. 2013), pp. 1786–1794. ISSN: 0167-739X. DOI: 10.1016/j.future.2013.01.004.

[269] Pinar Alper et al. "LabelFlow Framework for Annotating Workflow Provenance". In: *Informatics* 5.1 (Feb. 2018), p. 11. DOI: 10.3390/informatics5010011.

[270] Carole Goble et al. "Bioschemas.org". In: Poster. F1000Research, 2017. DOI: 10.7490/f1000research.1114493.1.

[271] Daniel Garijo, Yolanda Gil, and Oscar Corcho. "Towards Workflow Ecosystems through Semantic and Standard Representations". In: *2014 9th Workshop on Workflows in Support of Large-Scale Science*. IEEE, Nov. 2014, pp. 94–104. ISBN: 978-1-4799-7067-4. DOI: 10.1109/{WORKS}.2014.13.

[272] Peter Sefton et al. "DataCrate: a method of packaging, distributing, displaying and archiving Research Objects". In: (July 2018). Workshop on Research Objects (RO2018) at IEEE eScience 2018. DOI: 10.5281/zenodo.1312323.

[273] Christopher Woods. *BioExcel Webinar #28: BioSimSpace –filling the gaps between molecular simulation codes*. Accessed December 2018. June 2018. URL: https://youtu.be/pD8mhj3WElE?t=1599.

[274] *BioSimSpace*. Accessed November 2018. 2018. URL: https://biosimspace.org/.

[275] Alex L Mitchell et al. "EBI Metagenomics in 2017: enriching the analysis of microbial communities, from sequence reads to assemblies". In: *Nucleic Acids Research* 46.D1 (2018), pp. D726–D735. DOI: 10.1093/nar/gkx967.

[276] *Data on the Web Best Practices*. Accessed September 2018. URL: https://www.w3.org/TR/dwbp/#ReuseVocabularies.

[277] Gaurav Kaushik et al. "Rabix: an open-source workflow executor supporting recomputability and interoperability of workflow descriptions." In: *Pac Symp Biocomput* 22 (2017), pp. 154–165. DOI: 10.1142/9789813207813_0016.

[278] SEAN R. EDDY. "A New Generation of Homology Search Tools Based on Probabilistic Inference". In: *Genome Informatics 2009*. Oct. 2009. DOI: 10.1142/9781848165632_0019.

[279] *ebi-metagenomics-cwl/hmmsearch.cwl at master ·FarahZKhan/ebi − metagenomics − cwl*. Accessed October 2018. URL: https://github.com/FarahZKhan/ebi-metagenomics-cwl/blob/master/tools/hmmsearch.cwl.

[280] *TopMed RNA-seq workflow*. Accessed September 2018. 2018. URL: https://github.com/FarahZKhan/cwl_workflows/blob/cwlprov_testing/topmed-workflows/TOPMed_RNAseq_pipeline/rnaseq_pipeline_fastq.cwl.

[281] Mark Robinson et al. "Common Workflow Language Viewer". In: Accessed August 2017. July 2017. URL: https://view.commonwl.org/.

[282] *researchobject.org*. Accessed September 2018. URL: http://www.researchobject.org/overview/.

[283] Kyle Chard et al. "I'll take that to go: Big data bags and minimal identifiers for exchange of large, complex datasets". In: IEEE, Dec. 2016, pp. 319–328. ISBN: 978-1-4673-9005-7. DOI: 10.1109/bigdata.2016.7840618. (Accessed July 13, 2018).

[284] Luc Moreau et al. *PROV-N: The Provenance Notation*. W3C, Apr. 2013. (Accessed Aug. 16, 2017).

[285] Trung Dong Huynh et al. *The PROV-JSON Serialization. A JSON Representation for the PROV Data Model*. Tech. rep. Apr. 2013. URL: http://www.w3.org/Submission/2013/SUBM-prov-json-20130424/.

[286] James Manger. *Named Information*. Accessed October 2018. 2016. URL: https://www.iana.org/assignments/named-information/named-information.xhtml.

[287] EMC Education Services. *Information Storage and Management: Storing, Managing, and Protecting Digital Information in Classic, Virtualized, and Cloud Environments, Second Edition*. Ed. by Somasundaram Gnanasundaram and Alok Shrivastava. ISBN 978-1-118-09483-9. John Wiley & Sons, Apr. 2012. ISBN: 9781118094839.

[288] *JSON-LD 1.0*. Accessed September 2018. 2014. URL: https://www.w3.org/TR/json-ld/.

[289] Stian Soiland-Reyes, Matthew Gamble, and Robert Haines. "Research Object Bundle 1.0". In: (2014). URL: https://zenodo.org/record/12586.

[290] *common-workflow-language/cwlprov: Profile for provenance research object of a CWL workflow run*. Accessed September 2018. 2018. URL: https://w3id.org/cwl/prov.

[291] *The Wfdesc Ontology*. Accessed September 2018. 2016. URL: http://wf4ever.github.io/ro/2016-01-28/wfdesc/.

[292] *The Wfprov Ontology*. Accessed September 2018. 2016. URL: http://wf4ever.github.io/ro/2016-01-28/wfprov/.

[293] Yang Cao et al. "ProvONE: Extending PROV to support the DataONE scientific community". In: (June 2016). URL: http://homepages.cs.ncl.ac.uk/paolo.missier/doc/dataone-prov-3-years-later.pdf.

[294] Stian Soiland-Reyes and Marcos Cáceres. "The Archive and Package (arcp) URI scheme". In: (July 2018). Preprint, to appear in Proceedings of 2018 IEEE 14th International Conference on e-Science (e-Science). DOI: 10.5281/zenodo.1320264. URL: http://s11.no/2018/arcp.html.

[295] *PROV-N: The Provenance Notation*. Accessed September 2018. 2013. URL: https://www.w3.org/TR/prov-n/#expression-Agent.

[296] *prov 1.5.2*. Accessed September 2018. 2018. URL: https://pypi.org/project/prov/.

[297] *common-workflow-language/cwltool:Common Workflow Language reference implementation*. Accessed September 2018. 2016. URL: https://github.com/common-workflow-language/cwltool#cwl-tool-control-flow.

[298] Centre for Genomic Regulation (CRG). *Nextflow: Tracing and visualization*. Accessed November 2018. 2018. URL: https://www.nextflow.io/docs/latest/tracing.html#trace-report.

[299] Tazro Ohta, Tomoya Tanjo, and Osamu Ogasawara. "Accumulating computational resource usage of genomic data analysis workflow to optimize cloud computing instance selection". In: *bioRxiv* (2018). DOI: 10.1101/456756.

[300] A Cristofori et al. "Usage Record–Format Recommendation". In: *Open Grid Forum*. GFD-R-P.204. 2013. URL: https://www.ogf.org/documents/GFD.204.pdf.

[301] *interoperability — Definition of interoperability in English by Oxford Dictionaries*. Accessed September 2018. URL: https://en.oxforddictionaries.com/definition/interoperability.

[302] A. Tolk. "What Comes After the Semantic Web - PADS Implications for the Dynamic Web". In: *20th Workshop on Principles of Advanced and Distributed Simulation (PADS'06)*. IEEE. DOI: 10.1109/pads.2006.39.

[303] Alexander Dobin and Thomas R Gingeras. "Mapping RNA-seq reads with STAR". In: *Current protocols in bioinformatics* 51.1 (2015), pp. 11–14. DOI: 10.1002/0471250953.bi1114s51.

[304] Ana Conesa et al. "A survey of best practices for RNA-seq data analysis". In: *Genome Biology* 17.1 (Jan. 2016). DOI: 10.1186/s13059-016-0881-8.

[305] *heliumdatacommons*. Accessed September 2018. 2017. URL: https://github.com/heliumdatacommons.

[306] *Data Commons — NIH Common Fund*. Accessed September 2018. 2018. URL: https://commonfund.nih.gov/commons.

[307] Mark D Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship." In: *Sci Data* 3 (Mar. 2016), p. 160018. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.18.

[308] *heliumdatacommons/cwl_workflows: Example CWL Workflows that run on team Helium PIVOT architecture.* Accessed October 2018. URL: https://github.com/heliumdatacommons/cwl_workflows.

[309] *Trans-Omics for Precision Medicine (TOPMed) Program — National Heart, Lung, and Blood Institute (NHLBI)*. Accessed September 2018. 2014. URL: https://www.nhlbi.nih.gov/science/trans-omics-precision-medicine-topmed-program.

[310] *Gtex RNA-seq pipeline*. Accessed September 2018. 2017. URL: https://github.com/broadinstitute/gtex-pipeline/tree/master/rnaseq.

[311] Alexander Dobin et al. "STAR: ultrafast universal RNA-seq aligner". In: *Bioinformatics* 29.1 (Oct. 2012), pp. 15–21. DOI: 10.1093/bioinformatics/bts635. URL: https://doi.org/10.1093/bioinformatics/bts635.

[312] *Tool documentation: MarkDuplicates*. Accessed September 2018. URL: http://broadinstitute.github.io/picard/command-line-overview.html#MarkDuplicates.

[313] Heng Li et al. "The sequence alignment/map format and SAMtools". In: *Bioinformatics* 25.16 (2009), pp. 2078–2079.

[314] David S. DeLuca et al. "RNA-SeQC: RNA-seq metrics for quality control and process optimization". In: *Bioinformatics* 28.11 (Apr. 2012), pp. 1530–1532. DOI: 10.1093/bioinformatics/bts196.

[315] Bo Li and Colin N Dewey. "RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome". In: *BMC bioinformatics* 12.1 (2011), p. 323. DOI: 10.1186/1471-2105-12-323.

[316] Jeong-Sun Seo et al. "The transcriptional landscape and mutational profile of lung adenocarcinoma". In: *Genome Research* (2012). DOI: 10.1101/gr.145144.112.

[317] Chang Xu. "A review of somatic single nucleotide variant calling algorithms for next-generation sequencing data". In: *Computational and structural biotechnology journal* (2018). DOI: 10.1016/j.csbj.2018.01.003.

[318] *Alignment Workflow*. Accessed September 2018. URL: https://github.com/FarahZKhan/topmed-workflows/blob/cwlprov_testing/aligner/sbg-alignment-cwl/topmed-alignment.cwl.

[319] *Data Biosphere*. Accessed September 2018. 2018. URL: https://github.com/DataBioSphere.

[320] *statgen/docker-alignment: Dockerfile for Alignment*. Accessed September 2018. 2017. URL: https://github.com/statgen/docker-alignment.

[321] *Abecasis Lab - Genome Analysis Wiki*. Accessed September 2018. 2017. URL: https://genome.sph.umich.edu/wiki/Abecasis_Lab.

[322] Guy Cochrane et al. "Facing growth in the European Nucleotide Archive". In: *Nucleic Acids Research* 41.D1 (Nov. 2012), pp. D30–D35. DOI: 10.1093/nar/gks1175.

[323] Gregory G Faust and Ira M Hall. "SAMBLASTER: fast duplicate marking and structural variant read extraction". In: *Bioinformatics* 30.17 (2014), pp. 2503–2505.

[324] *Introduction to Variant Calling*. Accessed September 2018. 2014. URL: https://bioconductor.org/help/course-materials/2014/CSAMA2014/3_Wednesday/lectures/VariantCallingLecture.pdf.

[325] Christopher T. Saunders et al. "Strelka: accurate somatic small-variant calling from sequenced tumor–normal sample pairs". In: *Bioinformatics* 28.14 (May 2012), pp. 1811–1817. DOI: 10.1093/bioinformatics/bts271.

[326] *Blue Collar Bioinformatics*. Accessed September 2018. URL: http://bcb.io/.

[327] *Somatic Variant Calling Workflow*. Accessed September 2018. 2018. URL: https://github.com/FarahZKhan/bcbio_test_cwlprov/blob/master/somatic/somatic-workflow/main-somatic.cwl.

[328] *Common Workflow Language (CWL) –bcbio-nextgen 1.1.0 documentation*. Accessed September 2018. 2017. URL: https://bcbio-nextgen.readthedocs.io/en/latest/contents/cwl.html#current-status.

[329] Farah Zaib Khan and Stian Soiland-Reyes. *CWL run of RNA-seq Analysis Workflow (CWLProv 0.5.0 Research Object)*. 2018. DOI: 10.17632/xnwncxpw42.1.

[330] Farah Zaib Khan and Stian Soiland-Reyes. *CWL run of Alignment Workflow (CWLProv 0.6.0 Research Object)*. 2018. DOI: 10.17632/6wtpgr3kbj.1.

[331] Farah Zaib Khan and Stian Soiland-Reyes. *CWL run of Somatic Variant Calling Workflow (CWLProv 0.5.0 Research Object)*. 2018. DOI: 10.17632/97hj93mkfd.3.

[332] Stian Soiland-Reyes and Farah Zaib Khan. *common-workflow-language/cwlprov-py: cwlprov-py 0.1.1*. Oct. 2018. DOI: 10.5281/zenodo.1471376. URL: https://doi.org/10.5281/zenodo.1471376.

[333] Lucian Carata et al. "A primer on provenance". In: *Communications of the ACM* 57.5 (May 2014), pp. 52–60. DOI: 10.1145/2596628.

[334] Donghoon Kim and Mladen A Vouk. "Assessing Run-time Overhead of Securing Kepler". In: *Procedia Computer Science* 80 (2016), pp. 2281–2286. DOI: 10.1016/j.procs.2016.05.412.

[335] Carole Goble. "Better Software, Better Research". In: *IEEE Internet Computing* 18.5 (Sept. 2014), pp. 4–8. DOI: 10.1109/mic.2014.88.

[336] *Software Carpentry*. Accessed October 2018. URL: https://software-carpentry.org/.

[337] *Code Is Science*. Accessed October 2018. URL: http://www.codeisscience.com/.